(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号
特開2002－23622
（P2002－23622A）
(43)公開日　平成14年１月23日(2002.1.23)

(51)Int.Cl.⁷	識別記号	ＦＩ	ﾃｰﾏｺｰﾄﾞ(参考)
Ｇ０９Ｃ　1/00	６１０	Ｇ０９Ｃ　1/00	６１０Ｂ　５Ｊ１０４

審査請求　未請求　請求項の数18　ＯＬ　（全 22 頁）

(21)出願番号　特願2000－211686(P2000－211686)

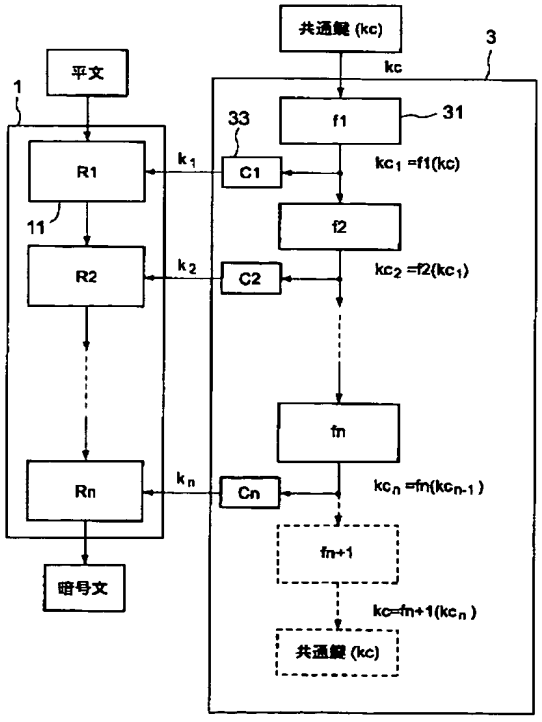
(22)出願日　平成12年７月12日(2000.7.12)

(71)出願人　000003078
株式会社東芝
東京都港区芝浦一丁目１番１号
(72)発明者　村谷　博文
神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内
(72)発明者　本山　雅彦
神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内
(74)代理人　100058479
弁理士　鈴江　武彦　（外６名）

最終頁に続く

(54)【発明の名称】　暗号化装置、復号装置及び拡大鍵生成装置、拡大鍵生成方法並びに記録媒体

(57)【要約】
【課題】　拡大鍵生成のための遅延時間の発生を回避し且つＯｎ－ｔｈｅ－ｆｌｙの鍵生成を可能とした暗号化装置を提供すること。
【解決手段】　暗号化時と復号時とで拡大鍵の使用順が逆になる共通鍵ブロック暗号方式の暗号化装置の拡大鍵生成部３において、初段からの段数と、最終段からの段数とが等しい２つのラウンド関数ｆ１とｆｎ＋１とを、互いに逆関数になるように設定する。これによって、暗号化時にも復号時にも共通鍵を入力として直ちに且つ使用順に拡大鍵を逐次生成していくことが可能になる。また、暗号化時の拡大鍵生成と復号時の拡大鍵生成とが基本的に同一になる。



【特許請求の範囲】

【請求項１】暗号化時のデータ撹拌処理と復号時のデータ撹拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による暗号化装置であって、

複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、２段目以降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するラウンド処理手段と、

前記ラウンド処理手段の全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力するための出力手段とを備え、

前記ラウンド処理手段は、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施すものであることを特徴とする暗号化装置。

【請求項２】前記ラウンド関数系列は、初段からの段数と最終段からの段数とが一致する２つのラウンド関数を互いに逆関数になるように設定したものであることを特徴とする請求項１に記載の暗号化装置。

【請求項３】前記ラウンド関数系列は、少なくとも第１の特定の段と第２の特定の段との間の連続する複数段について、該第１の特定の段からの段数と該第２の特定の段からの段数とが一致する２つのラウンド関数を互いに逆関数になるように設定した部分系列を含むものであることを特徴とする請求項１に記載の暗号化装置。

【請求項４】前記ラウンド関数系列は、少なくとも第１の特定の段から段数増加方向へ特定段数隔てた段までの連続する範囲と第２の特定の段から段数減少方向へ特定段数隔てた段までの連続する範囲について、該第１の特定の段からの段数と該第２の特定の段からの段数とが一致する２つのラウンド関数を互いに逆関数になるように設定した部分系列を含むものであることを特徴とする請求項１に記載の暗号化装置。

【請求項５】前記ラウンド関数系列を、請求項３または４に記載の部分系列を含まないように設定したことを特徴とする請求項１に記載の暗号化装置。

【請求項６】前記出力手段は、前記拡大鍵の出力のために前記中間状態を用いる際に、該中間状態については、その全ビットのうちから選択した当該中間状態を一意に決定するには十分ではない部分のみを用いることを特徴とする請求項１ないし５のいずれか１項に記載の暗号化装置。

【請求項７】前記ラウンド関数系列に属するラウンド関数のうち、少なくとも、その初段若しくは初段から段数増加方向へ所定段数隔てた段までの連続する範囲、及び又は最終段若しくは最終段から段数減少方向へ所定段数隔てた段までの連続する範囲に属するラウンド関数に対応する中間状態は、前記拡大鍵として又はそのもととなるデータとして用いないことを特徴とする請求項１ないし６のいずれか１項に記載の暗号化装置。

【請求項８】前記ラウンド関数系列に属するラウンド関数のうち、少なくとも、その初段からの段数と最終段からの段数とが一致する２つのラウンド関数に対応する中間状態のいずれか一方又は両方は、前記拡大鍵として又はそのもととなるデータとして用いないことを特徴とする請求項１ないし７のいずれか１項に記載の暗号化装置。

【請求項９】前記ラウンド関数系列に属するラウンド関数のうち、少なくとも、その初段若しくは初段から段数増加方向へ所定段数隔てた段までの連続する範囲、及び又は最終段若しくは最終段から段数減少方向へ所定段数隔てた段までの連続する範囲に属するラウンド関数に対応する中間状態のうち、初段からの段数と最終段からの段数とが一致する２つのラウンド関数のいずれか一方又は両方に対応する中間状態は、前記拡大鍵として又はそのもととなるデータとして用いないことを特徴とする請求項１ないし６のいずれか１項に記載の暗号化装置。

【請求項１０】複数の前記拡大鍵のうちの任意のものが常には一致しないようにしたことを特徴とする請求項１ないし９のいずれか１項に記載の暗号化装置。

【請求項１１】複数の前記拡大鍵のうちの任意のものが、それら拡大鍵の全ビットのうちの任意のビット群についても、常には一致しないようにしたことを特徴とする請求項１０に記載の暗号化装置。

【請求項１２】前記ラウンド処理手段及び前記出力手段は、前記データ撹拌処理に必要な拡大鍵数を越える数の拡大鍵を、該データ撹拌処理に提供可能であり、

前記提供可能な拡大鍵のうち実際に前記データ撹拌処理に提供すべき拡大鍵を示す情報、又は前記データ撹拌処理に提供すべき拡大鍵及びその提供する順番を示す情報を拡張共通鍵とし、

前記出力手段は、前記拡張共通鍵に従って、前記拡大鍵を出力することを特徴とする請求項１ないし１１のいずれか１項に記載の暗号化装置。

【請求項１３】暗号化時のデータ撹拌処理と復号時のデータ撹拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による復号装置であって、

複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、２段目以降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するラウンド処理手段と、

前記ラウンド処理手段の全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット

又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力するための出力手段とを備え、

前記ラウンド処理手段は、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施すものであることを特徴とする復号装置。

【請求項１４】暗号化時のデータ撹拌処理と復号時のデータ撹拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による暗号化装置又は復号装置に用いられる拡大鍵生成装置であって、

複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、２段目以降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するラウンド処理手段と、

前記ラウンド処理手段の全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力するための出力手段とを備え、

前記ラウンド処理手段は、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施すものであることを特徴とする拡大鍵生成装置。

【請求項１５】暗号化時のデータ撹拌処理と復号時のデータ撹拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による暗号化装置のための拡大鍵生成方法であって、

複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、２段目以降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するとともに、

全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力するステップを有し、

前記中間状態の生成にあたっては、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施すことを特徴とする拡大鍵生成方法。

【請求項１６】暗号化時のデータ撹拌処理と復号時のデータ撹拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による復号装置のための拡大鍵生成方法であって、

複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、２段目以降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するとともに、

全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力するステップを有し、

前記中間状態の生成にあたっては、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施すことを特徴とする拡大鍵生成方法。

【請求項１７】暗号化時のデータ撹拌処理と復号時のデータ撹拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による暗号化装置のための拡大鍵生成プログラムを記録したコンピュータ読取り可能な記録媒体であって、

複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、２段目以降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するとともに、全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力させ、

前記中間状態の生成にあたっては、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施させるためのプログラムを記録したコンピュータ読取り可能な記録媒体。

【請求項１８】暗号化時のデータ撹拌処理と復号時のデータ撹拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による復号装置のための拡大鍵生成プログラムを記録したコンピュータ読取り可能な記録媒体であって、

複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、２段目以降では、前段にて生成された中間状態を入

力として所定のラウンド関数を施して新たな中間状態を生成するとともに、全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力させ、

前記中間状態の生成にあたっては、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施させるためのプログラムを記録したコンピュータ読取り可能な記録媒体。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】本発明は、暗号化時と復号時とで複数の拡大鍵を逆の順番で用いる暗号化装置、復号装置及び拡大鍵生成装置、拡大鍵生成方法並びに記録媒体に関する。

【０００２】

【従来の技術】電子化された情報、とりわけ著作権に係る情報や機密情報やプライバシーに係る情報等のセキュリティ・コントロールのために、暗号技術の重要性が非常に高くなっている。実際に暗号技術は様々な分野において様々な形で利用されている。

【０００３】暗号方式には様々なものがあるが、そのうちの一つに、共通鍵暗号方式がある。共通鍵暗号方式は、暗号化の際に用いられた鍵と同一の鍵（共通鍵、秘密鍵）を用いて復号が行われる方式である。

【０００４】共通鍵暗号方式にも種々のものがあるが、そのうちの一つに、拡大鍵を用いる方式がある。この方式では、共通鍵をもとに、それが持つビット数より多い総ビット数の複数の拡大鍵を生成する。

【０００５】拡大鍵の生成方式の一つに、共通鍵に対してラウンド関数（段関数）を作用させ、その出力値をもとに拡大鍵を生成とするとともに、さらに該出力値にラウンド関数を作用させ、その出力値をもとに次の拡大鍵を生成とするとともに、さらに該出力値にラウンド関数を作用させ…、というように、次々とラウンド関数を作用させて拡大鍵を逐次的に生成していくものがある。このような方式をここではラウンド方式と呼ぶものとする。

【０００６】なお、このような拡大鍵生成方式をとる共通鍵暗号方式としては、例えば、共通鍵ブロック暗号方式がある。なお、共通鍵ブロック暗号方式は、データ撹拌部についても、処理単位となる所定ビット長のブロック・データに、ラウンド関数を次々と作用させて暗号化または復号を行う構造を有するものであり、その代表的な基本構造にＳＰＮ型とＦｅｉｓｔｅｌ型等がある。

【０００７】さて、拡大鍵の生成にラウンド方式をとる場合には、例えばブロック暗号のように、暗号化の際に用いられた順番とは逆の順番で拡大鍵を用いることが要求される。

【０００８】以下、このような方式の問題点について説明する。

【０００９】図３８に、従来の暗号化装置の拡大鍵生成部の構成例を示す。

【００１０】まず、データ撹拌部では暗号化処理に拡大鍵（１）を必要とする。そこで、共通鍵にラウンド関数（１）を作用させ、その出力値を求め、これに拡大鍵変換（１）を作用させて、拡大鍵（１）を得る。データ撹拌部は、この拡大鍵（１）を用いて暗号化処理を行う。

【００１１】次に、データ撹拌部では暗号化処理に拡大鍵（２）を必要とする。そこで、ラウンド関数（１）の出力値にラウンド関数（２）を作用させ、その出力値を求め、これに拡大鍵変換（２）を作用させて、拡大鍵（２）を得る。データ撹拌部は、この拡大鍵（２）を用いて暗号化処理を行う。

【００１２】以降、同様にして、拡大鍵生成部による拡大鍵の生成と、データ撹拌部による暗号化処理が行われる。

【００１３】ここで、復号時の処理を考える。

【００１４】復号時には、暗号時とは逆の順、すなわち、拡大鍵（ｎ）→拡大鍵（１）の順番に、拡大鍵を用いる必要がある。ところが、図３８と同様の構成の拡大鍵生成部を有する従来の復号装置では、拡大鍵は、拡大鍵（１）→拡大鍵（ｎ）の順に生成されるので、例えば、データ撹拌部の処理に先立って、すべての拡大鍵を生成し、メモリに記憶しておく必要があった。

【００１５】しかしながら、例えばＩＣカードのように貧弱なハードウェア環境しかない装置では、復号に必要な拡大鍵をすべて格納するだけの記憶領域の余裕がないという問題点がある。

【００１６】一方、この問題を回避するために、図３９に例示するような構成が考えられる。この従来の復号装置の拡大鍵生成部の構成例では、一旦、暗号化時と同一の拡大鍵生成処理を行って最終ラウンドでラウンド関数を作用させて得られる出力値（１０２０）を求める。その後、あらためて、該出力値（１０２０）に暗号化時とは逆のラウンド方向に各ラウンド関数の逆関数を作用させて、拡大鍵（ｎ）→拡大鍵（１）の順に、すなわちＯｎ－ｔｈｅ－ｆｌｙに拡大鍵を生成していく。

【００１７】しかしながら、最初に暗号化時と同一の拡大鍵生成処理を行う不要な時間のために、復号が開始されるまでの遅延時間が発生するという問題点があった。

【００１８】

【発明が解決しようとする課題】以上説明したように、従来の技術では、拡大鍵を逆順に生成することはできないので、復号処理に先立って、全拡大鍵を生成し記憶しておく必要があるが、例えばＩＣカードのように貧弱な

ハードウェア環境では、復号に必要な拡大鍵をすべて格納するだけの記憶領域の余裕がないという問題点があった。

【００１９】また、Ｏｎ－ｔｈｅ－ｆｌｙの鍵生成によって、この問題を回避するためには、一旦、暗号化時と同一の拡大鍵生成処理を行って最終ラウンドでラウンド関数を作用させて得られる出力値を求めた後に、あらためて該出力値に逆のラウンド方向に各ラウンド関数の逆関数を作用させていくことが必要になるが、この場合も、復号が開始されるまでの遅延時間が避けられないという問題点があった。

【００２０】本発明は、上記事情を考慮してなされたもので、拡大鍵生成のための遅延時間の発生を回避しもしくは小さくし、かつ、Ｏｎ－ｔｈｅ－ｆｌｙの鍵生成を可能とした暗号化装置、復号装置及び拡大鍵生成装置、拡大鍵生成方法並びに記録媒体を提供することを目的とする。

【００２１】

【課題を解決するための手段】本発明は、暗号化時のデータ撹拌処理と復号時のデータ撹拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による暗号化装置または復号装置であって、複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、２段目以降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するラウンド処理手段と、前記ラウンド処理手段の全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力するための出力手段とを備え、前記ラウンド処理手段は、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施すものであることを特徴とする。

【００２２.】好ましくは、前記ラウンド関数系列は、初段からの段数と最終段からの段数とが一致する２つのラウンド関数を互いに逆関数になるように設定したものであるようにしてもよい。また、好ましくは、前記ラウンド関数系列は、少なくとも第１の特定の段と第２の特定の段との間の連続する複数段について、該第１の特定の段からの段数と該第２の特定の段からの段数とが一致する２つのラウンド関数を互いに逆関数になるように設定した部分系列を含むものであるようにしてもよい。また、好ましくは、前記ラウンド関数系列は、少なくとも第１の特定の段から段数増加方向へ特定段数隔てた段までの連続する範囲と第２の特定の段から段数減少方向へ特定段数隔てた段までの連続する範囲について、該第１の特定の段からの段数と該第２の特定の段からの段数とが一致する２つのラウンド関数を互いに逆関数になるように設定した部分系列を含むものであるようにしてもよい。

【００２３】好ましくは、前記出力手段は、前記拡大鍵の出力のために前記中間状態を用いる際に、該中間状態については、その全ビットのうちから選択した当該中間状態を一意に決定するには十分ではない部分のみを用いるようにしてもよい。

【００２４】好ましくは、前記ラウンド関数系列に属するラウンド関数のうち、少なくとも、その初段若しくは初段から段数増加方向へ所定段数隔てた段までの連続する範囲、及び又は最終段若しくは最終段から段数減少方向へ所定段数隔てた段までの連続する範囲に属するラウンド関数に対応する中間状態は、前記拡大鍵として又はそのもととなるデータとして用いないようにしてもよい。また、好ましくは、前記ラウンド関数系列に属するラウンド関数のうち、少なくとも、その初段からの段数と最終段からの段数とが一致する２つのラウンド関数に対応する中間状態のいずれか一方又は両方は、前記拡大鍵として又はそのもととなるデータとして用いないようにしてもよい。また、好ましくは、前記ラウンド関数系列に属するラウンド関数のうち、少なくとも、その初段若しくは初段から段数増加方向へ所定段数隔てた段までの連続する範囲、及び又は最終段若しくは最終段から段数減少方向へ所定段数隔てた段までの連続する範囲に属するラウンド関数に対応する中間状態のうち、初段からの段数と最終段からの段数とが一致する２つのラウンド関数のいずれか一方又は両方に対応する中間状態は、前記拡大鍵として又はそのもととなるデータとして用いないようにしてもよい。

【００２５】好ましくは、複数の前記拡大鍵のうちの任意のものが常には一致しないようにしてもよい。また、好ましくは、複数の前記拡大鍵のうちの任意のものが、それら拡大鍵の全ビットのうちの任意のビット群についても、常には一致しないようにしてもよい。

【００２６】好ましくは、前記ラウンド処理手段及び前記出力手段は、前記データ撹拌処理に必要な拡大鍵数を越える数の拡大鍵を、該データ撹拌処理に提供可能であり、前記提供可能な拡大鍵のうち実際に前記データ撹拌処理に提供すべき拡大鍵を示す情報、又は前記データ撹拌処理に提供すべき拡大鍵及びその提供する順番を示す情報を拡張共通鍵とし、前記出力手段は、前記拡張共通鍵に従って、前記拡大鍵を出力するようにしてもよい。

【００２７】また、本発明は、暗号化時のデータ撹拌処理と復号時のデータ撹拌処理とで逆の順番で複数の拡大鍵を使用する共通鍵暗号方式による暗号化装置又は復号装置に用いられる拡大鍵生成装置であって、複数段のラウンド関数について、初段では、共通鍵を入力として所定のラウンド関数を施して中間状態を生成し、２段目以

降では、前段にて生成された中間状態を入力として所定のラウンド関数を施して新たな中間状態を生成するラウンド処理手段と、前記ラウンド処理手段の全部又は一部の段にて生成された前記中間状態の各々について、該中間状態の全ビット又はその一部をそのまま又はこれに所定の変換処理を施した後に前記拡大鍵として出力するための出力手段とを備え、前記ラウンド処理手段は、複数のラウンド関数を従属接続したラウンド関数系列であって前記共通鍵をその初段へ入力した場合にその最終段が該共通鍵と同一の値を生成するように設定されたラウンド関数系列における全段又はそのうちの一部で初段から連続した複数段についてのラウンド関数を、該ラウンド関数系列の段の順番に従って施すものであることを特徴とする。

【００２８】なお、暗号化装置に係る本発明は、暗号化方法、復号装置、復号方法、拡大鍵生成装置又は拡大鍵生成方法に係る発明としても成立する。また、それら発明は、コンピュータに当該発明に相当する手順を実行させるための（あるいはコンピュータを当該発明に相当する手段として機能させるための、あるいはコンピュータに当該発明に相当する機能を実現させるための）プログラムを記録したコンピュータ読取り可能な記録媒体としても成立する。

【００２９】本発明によれば、拡大鍵生成のためのラウンド関数の系列を、共通鍵を入力し、共通鍵と同じ値を出力するように設定することによって、暗号化時と復号時の両方において、従来のような不必要な遅延時間や記憶容量の消費なく、共通鍵からＯｎ－ｔｈｅ－ｆｌｙに拡大鍵を生成することが可能になる。

【００３０】

【発明の実施の形態】以下、図面を参照しながら発明の実施の形態を説明する。

【００３１】本発明は、暗号化時と復号時とで逆の順番に拡大鍵を用いる共通鍵暗号方式のすべてに適用可能であるが、以下では、所定ビット長のブロック・データに対して逐次的に各拡大鍵を用いたデータ撹拌処理を行っていくような共通鍵ブロック暗号方式を例にとって説明する。

【００３２】まず、本実施形態の基本的な構成例について説明する。

【００３３】なお、以下で参照する各図では（当該暗号に着目して説明するため）暗号対象となるデータを平文として示してあるが、もちろん、暗号対象となるデータがすでに同一のまたは他の暗号方式によって暗号化されたものであってもよい。

【００３４】また、本暗号方式は、ハードウェアによってもソフトウェアによっても実現可能であり、以下に示す構成例は、暗号化装置（復号装置）の機能ブロック図としても成立し、また暗号アルゴリズム（復号アルゴリズム）の機能モジュール図もしくはフローチャート図としても成立する。

【００３５】図１に、本発明の一実施形態に係る暗号化装置の構成例を示す。

【００３６】図１に示されるように、本暗号化装置は、データ撹拌部１と、拡大鍵生成部３を備えている。

【００３７】拡大鍵生成部３は、複数のラウンド処理部３１を備えている。

【００３８】ｆ１で示したラウンド処理部は、共通鍵ｋｃにラウンド関数ｆ１を作用させて中間状態$kc_1 = f1(kc)$を出力する。ｆ２で示したラウンド処理部は、前段のラウンド処理部の出力した中間状態kc_1にラウンド関数ｆ２を作用させて中間状態$kc_2 = f2(kc_1) = f2(f1(kc))$を出力する。図示しないｆ３～ｆn-1に対応するラウンド処理部も同様である。ｆｎで示したラウンド処理部は、前段のラウンド処理部の出力した中間状態kc_{n-1}にラウンド関数ｆｎを作用させて中間状態$kc_n = fn(kc_{n-1}) = fn(fn\text{-}1(\cdots f2(f1(kc))\cdots))$を出力する。

【００３９】ここで、本実施形態では、ｆn+1で示したラウンド処理部にて前段のラウンド処理部の出力した中間状態kc_nにラウンド関数ｆn+1を作用させ、これによって得られた出力値$kc_n = fn+1(kc_{n-1}) = fn+1(fn(fn\text{-}1(\cdots f2(f1(kc))\cdots)))$が、共通鍵ｋｃに等しくなるようする。また、このラウンド関数ｆn+1の逆関数$fn+1^{-1}$が、復号装置における拡大鍵生成部の初段のラウンド処理部のラウンド関数になる。なお、図１の構成例では、この暗号化装置における拡大鍵生成部にはラウンド関数ｆn+1のラウンド処理部を備えなくても構わない（備えても構わない）。

【００４０】また、拡大鍵生成部３は、複数の拡大鍵変換部３３を備えている。

【００４１】ｃ１で示した拡大鍵変換部は、ｆ１で示したラウンド処理部の出力kc_1の全部または一部に拡大鍵変換関数ｃ１を作用させて、共通鍵ｋ１を生成する。ｃ２で示した拡大鍵変換部は、ｆ２で示したラウンド処理部の出力kc_2の全部または一部に拡大鍵変換関数ｃ２を作用させて、共通鍵ｋ２を生成する。図示しないｃ３～ｃn-1に対応するラウンド処理部も同様である。ｃｎで示した拡大鍵変換部は、ｆｎで示したラウンド処理部の出力kc_nの全部または一部に拡大鍵変換関数ｃｎを作用させて、共通鍵ｋｎを生成する。

【００４２】データ撹拌部１は、ここでは、従属接続された複数の（例えばラウンド関数による）撹拌処理部１１を備えるものとしている。

【００４３】Ｒ１で示した撹拌処理部は、暗号化の対象となるブロック・データを入力し、拡大鍵ｋ１を用いて、撹拌処理Ｒ１を行う。Ｒ２で示した撹拌処理部は、Ｒ２で示した撹拌処理部から出力されたブロック・データを入力し、拡大鍵ｋ２を用いて、撹拌処理Ｒ２を行う。図示しないＲ３～Ｒn-1に対応する撹拌処理部も同

様である。Ｒｎで示した攪拌処理部は、Ｒｎ-1 で示した攪拌処理部から出力されたブロック・データを入力し、拡大鍵ｋｎを用いて、攪拌処理Ｒｎを行う。Ｒｎで示した攪拌処理部の出力が、求めるべき暗号文となる。

【００４４】なお、複数のラウンド関数は、すべて異なるものであっても、すべて同じものであっても、異なるものと同じものが混在するものであってもよい。複数のラウンド関数を異なるものにする場合に、関数を異ならせる方法の他に、基本的には同じ関数であるが段に応じて異なる定数に依存するものにする方法などがある。また、複数のラウンド関数は、すべて線形関数であっても任意の関数であってもよいが、それらのうちの少なくとも一つを非線形関数にするのが好ましい。また、２以上のラウンド関数あるいは全てのラウンド関数を非線形関数にしてもよい。また、ラウンド関数は、変換テーブルを用いて実現する方法、行列演算やその他の演算で実現する方法、実回路によって実現する方法など、種々の構成方法がある。これらの点は、複数の拡大鍵変換関数についても同様である。なお、拡大鍵生成部３として、入力された中間状態またはその一部を、そのまま拡大鍵として出力する構成にする（あるいは、中間状態をデータ攪拌部１（または後述するスイッチング回路１５）に直結する）ことも可能である。

【００４５】なお、ブロック・データのデータ長と、共通鍵ｋｃのデータ長とは、同じであってもよいし、異なるものであってもよい。また、拡大鍵のデータ長と、ブロック・データのデータ長とは、同じであってもよいし、異なるものであってもよい。また、中間状態のデータ長と、拡大鍵のデータ長とは、同じであってもよいし、異なるものであってもよい。

【００４６】図２に、本発明の一実施形態に係る復号装置の構成例を示す。

【００４７】図２に示されるように、本復号装置は、データ攪拌部２と、拡大鍵生成部４を備えている。図２の復号装置は、図１の暗号化装置の逆変換を行う機能を有するものである。

【００４８】拡大鍵生成部４は、複数のラウンド処理部４２を備えており、図１の暗号化装置の拡大鍵生成部２の複数のラウンド関数の各々の逆関数を、逆の順番で作用させるものである。

【００４９】ｆn+1 $^{-1}$で示したラウンド処理部は、共通鍵ｋｃ＝ｆn+1 （ｋｃn）＝ｆn+1（ｆｎ（ｆn-1 （…ｆ２（ｆ１（ｋｃ））…））））にラウンド関数ｆn+1 $^{-1}$を作用させて、中間状態ｋｃｎ＝ｆn+1 $^{-1}$（ｋｃ）＝ｆn+1 $^{-1}$（ｆn+1 （ｆｎ（ｆn-1 （…ｆ２（ｆ１（ｋｃ））…））））＝ｆｎ（ｆn-1 （…ｆ２（ｆ１（ｋｃ））…））を出力する。ｆｎ$^{-1}$で示したラウンド処理部は、前段のラウンド処理部の出力した中間状態ｋｃｎにラウンド関数ｆn+1 $^{-1}$を作用させて、ｋｃn-1 ＝ｆn-1 （…ｆ２（ｆ１（ｋｃ））…）を出力する。図示しないｆn-1 ～ｆ３に対応するラウンド処理部も同様である。ｆ２$^{-1}$で示したラウンド処理部は、前段のラウンド処理部の出力した中間状態ｋｃ２＝ｆ２（ｆ１（ｋｃ））にラウンド関数ｆ２$^{-1}$を作用させて、ｋｃ１＝ｆ１（ｋｃ）を出力する。

【００５０】ここで、本実施形態では、ｆ１で示したラウンド処理部にて前段のラウンド処理部の出力した中間状態ｋｃ２にラウンド関数ｆ１$^{-1}$を作用させ、これによって得られた出力値は共通鍵ｋｃに等しくなる。また、このラウンド関数ｆ１$^{-1}$の逆関数ｆ１が、暗号化装置における拡大鍵生成部の初段のラウンド処理部のラウンド関数になる。なお、図２の構成例では、この復号装置には拡大鍵生成部ではラウンド関数ｆ１$^{-1}$のラウンド処理部を備えなくても構わない（備えても構わない）。

【００５１】また、拡大鍵生成部３は、複数の拡大鍵変換部４４を備えている。この部分は、図１の暗号化装置の対応する部分と同じ処理内容となる。

【００５２】データ攪拌部２は、ここでは、従属接続された複数の（例えばラウンド関数による）攪拌処理部２２を備えるものとしている。Ｒｎ$^{-1}$で示した攪拌処理部は、復号の対象となるブロック・データを入力し、拡大鍵ｋｎを用いて、暗号化装置の攪拌処理Ｒｎの逆変換となる攪拌処理Ｒｎ$^{-1}$を行う。同様に、他の攪拌処理部も順次、前段の攪拌処理部から出力されるブロック・データを入力し、拡大鍵Ｋn-1 、…、またはｋ２を用いて、攪拌処理Ｒn-1 $^{-1}$、…、またはＲ２$^{-1}$を行う。Ｒ１$^{-1}$で示した攪拌処理部は、Ｒ２$^{-1}$で示した攪拌処理部から出力されたブロック・データを入力し、拡大鍵ｋ１を用いて、暗号化装置の攪拌処理Ｒ１の逆変換となる攪拌処理Ｒ１$^{-1}$を行う。Ｒ１$^{-1}$で示した攪拌処理部の出力によって、求めるべき復号結果となるブロック・データが与えられる。

【００５３】すなわち、復号時において、すぐに暗号化時とは逆の順番での拡大鍵の生成に入り、拡大鍵を次々と生成していくことができる。

【００５４】以上のように、暗号化時のラウンド関数の系列（ただし、最終段は備えないこともある）と、その逆関数となる復号時のラウンド関数の系列（ただし、最終段は備えないこともある）とについて、暗号化時の最終段の出力に相当する値がもとの共通鍵に一致するようにラウンド関数の系列を設定することによって、暗号化時と復号時の両方において、従来のような不必要な遅延時間や記憶容量の消費なく、共通鍵からＯｎ－ｔｈｅ－ｆｌｙに拡大鍵を生成することが可能になる。

【００５５】次に、図１／図２の暗号化装置や復号装置の拡大鍵生成部の複数のラウンド処理部によるラウンド関数の系列について説明する。なお、暗号化装置におけるラウンド関数の系列と、復号装置におけるラウンド関数の系列とは、逆関数の関係になるので、一方が決まれば、他方も決まることになる。ここでは、暗号化装置の

方を例にとって説明する。

【００５６】ラウンド関数の系列ｆ１，ｆ２，…，ｆｎ+１について、そのラウンド関数の系列の内容、あるいは各々の順番のラウンド関数の内容は、当該ラウンド関数の系列が全体として共通鍵を入力し共通鍵と同じ値を出力する条件を満たす範囲内において適宜設定可能であり、様々なバリエーションがある。以下、ラウンド関数の系列のバリエーションのいくつかを例示列挙的に説明する。

【００５７】（ラウンド・トリップ構成）まず、ラウンド・トリップ構成について説明する。

【００５８】ここでは、ラウンド関数の系列の段数を２ｒ段とする（なお、前述したように、２ｒ段目のラウンド関数は、備えられないことがある）。

【００５９】ラウンド関数の系列を構成する一つの方法は、０≦ｉ≦ｒを満たすすべてのｉについて、第（ｒ＋ｉ）段目の段関数を、第（ｒ－ｉ＋１）段目の段関数の逆関数になっている、という関係を満たすように構成する方法である。

【００６０】例えば、ラウンド関数の系列を８段、すなわち、

ｆ１、ｆ２、ｆ３、ｆ４、ｆ５、ｆ６、ｆ７、ｆ８

とし、ｆ１～ｆ４を任意のラウンド関数として、ｆ５＝ｆ４$^{-1}$、ｆ６＝ｆ３$^{-1}$、ｆ７＝ｆ２$^{-1}$、ｆ８＝ｆ１$^{-1}$とすると、

ｆ１、ｆ２、ｆ３、ｆ４、ｆ４$^{-1}$、ｆ３$^{-1}$、ｆ２$^{-1}$、ｆ１$^{-1}$

という順番の系列となる。すなわち、共通鍵を入力として、ｆ１，ｆ２，ｆ３，ｆ４、ｆ４$^{-1}$、ｆ３$^{-1}$、ｆ２$^{-1}$、ｆ１$^{-1}$を次々と作用させることによって、最終段の出力が共通鍵と一致するようになる。

【００６１】このような構成を、ラウンド・トリップ構成と呼ぶものとする。また、この様子を図３に概念的に示す。

【００６２】ラウンド・トリップ構成を採用した場合、暗号化装置におけるラウンド関数の系列と、復号装置におけるラウンド関数の系列とが同一になる。

【００６３】上記の例の場合、暗号化装置における８段のラウンド関数を、

ｆ１，ｆ２，ｆ３，ｆ４、ｆ４$^{-1}$、ｆ３$^{-1}$、ｆ２$^{-1}$、ｆ１$^{-1}$

とすると、復号装置における８段のラウンド関数は、この逆関数となるので、

（ｆ１$^{-1}$）$^{-1}$、（ｆ２$^{-1}$）$^{-1}$、（ｆ３$^{-1}$）$^{-1}$、（ｆ４$^{-1}$）$^{-1}$、（ｆ４）$^{-1}$、（ｆ３）$^{-1}$、（ｆ２）$^{-1}$、（ｆ１）$^{-1}$

であり、したがって、

ｆ１，ｆ２，ｆ３，ｆ４、ｆ４$^{-1}$、ｆ３$^{-1}$、ｆ２$^{-1}$、ｆ１$^{-1}$

となり、両者が一致することがわかる。

【００６４】なお、暗号化装置においても最終段のラウンド関数（上記の例では、ｆ１$^{-1}$）を備えなくてもよいし、復号装置においても最終段のラウンド関数（上記の例では、ｆ１）を備えなくてもよいが、いずれも最終段のラウンド関数を備えれば、構成が同一となるので、暗号化機能と復号機能の両方の機能を兼ね備えさせる装置においては、暗号化時と復号時とで１つの拡大鍵生成部を兼用することによって、装置規模を削減することも可能である。

【００６５】次に、この構成において、ラウンド関数の系列の前半の各ラウンド関数は、すべて異なるものであっても、すべて同じものであっても、異なるものと同じものが混在するものであってもよい。

【００６６】例えば、ラウンド関数の系列の前半の各ラウンド関数がすべて同じものである場合、段数を８段とすると、暗号側と復号側のいずれにおいても、

ｆ１、ｆ１、ｆ１、ｆ１、ｆ１$^{-1}$、ｆ１$^{-1}$、ｆ１$^{-1}$、ｆ１$^{-1}$

という系列になる。

【００６７】ところで、ラウンド・トリップ構成を採用した場合、ラウンド関数の系列において逆関数の関係にある対応部分の中間状態が同一になる。したがって、同一の中間状態に同一の拡大鍵変換関数を作用させると、同一の拡大鍵が生成される。そこで、これを避けるために、ラウンド関数の系列において逆関数の関係にある対応部分についての２つの拡大鍵変換部の拡大鍵変換関数として相異なるものを用いるようにしてもよい。

【００６８】例えば、８段のラウンド関数の系列を、

ｆ１，ｆ２，ｆ３，ｆ４、ｆ４$^{-1}$、ｆ３$^{-1}$、ｆ２$^{-1}$、ｆ１$^{-1}$

とし、ｆ１の出力を用いる拡大鍵変換関数をｃ１、…、ｆ２$^{-1}$の出力を用いる拡大鍵変換関数をｃ７とすると、拡大鍵変換関数ｃ１と拡大鍵変換関数ｃ７とを異ならせるようにしてもよい。ｃ２とｃ６、ｃ３とｃ５についても同様である。

【００６９】（ループ構成）次に、ループ構成について説明する。

【００７０】ラウンド・トリップ構成では、ラウンド関数の系列の後半を、その前半の逆関数としたが、ラウンド関数の系列の中の部分系列として、ラウンド・トリップ構成に相当する部分を全く持たない構成も可能である。

【００７１】このような構成を、ループ構成と呼ぶものとする。また、この様子を図４に概念的に示す。

【００７２】なお、ラウンド・トリップ構成では、ラウンド関数の系列の段数を偶数段としたが、ループ構成では、ラウンド関数の系列の段数は、偶数段であっても奇数段であってもよい。

【００７３】例えば、ラウンド関数の系列を８段とした場合に、共通鍵を入力として、ｆ１、ｆ２、ｆ３、ｆ

４、ｆ５、ｆ６、ｆ７、ｆ８を次々と作用させることによって、最終段の出力が共通鍵と一致するようにする。この場合、その逆関数は、

ｆ８-1、ｆ７-1、ｆ６-1、ｆ５-1、ｆ４-1、ｆ３-1、ｆ２-1、ｆ１-1

であり、共通鍵を入力すると、最終段の出力が共通鍵と一致することになる。

【００７４】また、例えば、ラウンド関数の系列の前半の各ラウンド関数がすべて同じものである場合、段数を８段とすると、暗号側では、

ｆ１、ｆ１、ｆ１、ｆ１、ｆ１、ｆ１、ｆ１、ｆ１

という系列になり、復号側では、

$f1^{-1}$、$f1^{-1}$、$f1^{-1}$、$f1^{-1}$、$f1^{-1}$、$f1^{-1}$、$f1^{-1}$、$f1^{-1}$

という系列になる。

【００７５】なお、このような条件を満たす関数は、シフト演算、行列演算、ガロア体演算など、種々のものがある。

【００７６】（ラウンド・トリップ／ループ複合構成）ラウンド関数の系列としては、その部分系列として、ラウンド・トリップ構成に相当する部分と、ループ構成に相当する部分とを複合的に備えるような構成も可能である。

【００７７】以下、ラウンド・トリップ構成部分を図３の表記方法で示し、ループ構成部分を図３の表記方法で示すものとして、図５～図９にいくつかのバリエーションを例示する。

【００７８】図５の例は、ラウンド・トリップ構成部分の途中にラウンド・トリップ構成部分を含むような入れ子構造になっているものである。図５の場合のラウンド関数の系列を例示すると、

$a1 \rightarrow a2 \rightarrow a3 \rightarrow b1 \rightarrow b2 \rightarrow b2^{-1} \rightarrow b1^{-1} \rightarrow a4 \rightarrow a5 \rightarrow a6 \rightarrow a6^{-1} \rightarrow a5^{-1} \rightarrow a4^{-1} \rightarrow a3^{-1} \rightarrow c1 \rightarrow c2 \rightarrow c2^{-1} \rightarrow d1 \rightarrow d1^{-1} \rightarrow c1^{-1} \rightarrow a2^{-1} \rightarrow a1^{-1}$

となる。この例の場合、$a1 \rightarrow a2 \rightarrow a3 \rightarrow a4 \rightarrow a5 \rightarrow a6 \rightarrow a6^{-1} \rightarrow a5^{-1} \rightarrow a4^{-1} \rightarrow a3^{-1} \rightarrow a2^{-1} \rightarrow a1^{-1}$というラウンド・トリップ構成の中に、$b1 \rightarrow b2 \rightarrow b2^{-1} \rightarrow b1^{-1}$というラウンド・トリップ構成と、$c1 \rightarrow c2 \rightarrow c2^{-1} \rightarrow c1^{-1}$というラウンド・トリップ構成が入っており、さらに、$c1 \rightarrow c2 \rightarrow c2^{-1} \rightarrow c1^{-1}$というラウンド・トリップ構成の中に、$d1 \rightarrow d1^{-1}$というラウンド・トリップ構成が入っている。

【００７９】図６の例は、ループ構成部分の途中にループ構成部分を含むような入れ子構造になっているものである。図６の場合のラウンド関数の系列を例示すると、

ｓ１→ｓ２→ｓ３→ｓ４→ｔ１→ｔ２→ｔ３→ｓ５→ｓ６→ｓ７→ｓ８

となる。この例の場合、ｓ１→ｓ２→ｓ３→ｓ４→ｓ５→ｓ６→ｓ７→ｓ８というループ構成の中に、ｔ１→ｔ２→ｔ３というループ構成が入っている。

【００８０】図７の例は、ループ構成部分の途中にラウンド・トリップ構成部分を含むような構造になっているものである。図７の場合のラウンド関数の系列を例示すると、

$s1 \rightarrow s2 \rightarrow s3 \rightarrow s4 \rightarrow s5 \rightarrow a1 \rightarrow a2 \rightarrow a3 \rightarrow a3^{-1} \rightarrow a2^{-1} \rightarrow a1^{-1} \rightarrow s6 \rightarrow s7 \rightarrow s8 \rightarrow s9$

となる。この例の場合、ｓ１→ｓ２→ｓ３→ｓ４→ｓ５→ｓ６→ｓ７→ｓ８→ｓ９というループ構成の中に、$a1 \rightarrow a2 \rightarrow a3 \rightarrow a3^{-1} \rightarrow a2^{-1} \rightarrow a1^{-1}$というラウンド・トリップ構成が入っている。

【００８１】図８の例は、ラウンド・トリップ構成部分の途中にループ構成部分を含むような構造になっているものである。図８の場合のラウンド関数の系列を例示すると、

$a1 \rightarrow a2 \rightarrow a3 \rightarrow a4 \rightarrow a5 \rightarrow a6 \rightarrow s1 \rightarrow s2 \rightarrow s3 \rightarrow s4 \rightarrow a6^{-1} \rightarrow a5^{-1} \rightarrow a4^{-1}\,a3^{-1} \rightarrow a2^{-1} \rightarrow a1^{-1}$

となる。この例の場合、$a1 \rightarrow a2 \rightarrow a3 \rightarrow a4 \rightarrow a5 \rightarrow a6 \rightarrow a6^{-1} \rightarrow a5^{-1} \rightarrow a4^{-1}\,a3^{-1} \rightarrow a2^{-1} \rightarrow a1^{-1}$というラウンド・トリップ構成の中に、ｓ１→ｓ２→ｓ３→ｓ４というループ構成が入っている。

【００８２】図９の例は、４つのラウンド・トリップ構成部分と、２つのループ部分を持つものである。

【００８３】もちろん、これら以外にも、ラウンド・トリップ構成部分とループ構成部分の組み合わせ方、あるいは階層構造の取り方など、種々のバリエーションが可能である。

【００８４】さて、図１／図２の構成例は、データ攪拌部で必要とする数の分だけ拡大鍵を生成することを想定したものであったが、データ攪拌部で必要とする数を越える数の分の拡大鍵を生成可能とするラウンド関数の段数を備え、生成可能な拡大鍵の一部をデータ攪拌部で使用する構成も可能である。

【００８５】この場合の図１／図２の暗号化装置／復号装置に対応する構成例を、図１０／図１１にそれぞれ示す。

【００８６】ここでは、図１／図２と相違する点を中心に説明する。もちろん、ラウンド関数の系列は、上記したラウンド・トリップ構成等を採用したものであってよい。

【００８７】図１０の５と図１１の６は、拡大鍵ｋｉと攪拌処理Ｒｊとの接続関係を示す部分であり、そのいくつかの具体例を図１２～図１５に例示している。なお、本実施形態では、図１０の５の接続関係と図１１の６の接続関係は同一になる。

【００８８】なお、ここでは、初段のラウンド関数への入力としての共通鍵と、最終段のラウンド関数からの出力されることになる共通鍵の双方または一方をも、中間状態として拡大鍵生成に利用してもよい。なお、後者を

用いる場合に、実際に最終段のラウンド関数の出力を用いるようにしてもよいし、共通鍵を記憶しておいてこれを用いるようにしてもよい。

【００８９】本実施形態では、生成可能な拡大鍵の数が、撹拌処理で必要な拡大鍵の数より多くなるように構成し、拡大鍵ｋｉと撹拌処理Ｒｊとを適宜対応付ける。なお、同一の拡大鍵を複数の撹拌処理に使用可能とする方法と、１つの拡大鍵を撹拌処理について排他的に使用可能とする方法とがある。

【００９０】なお、使用しないことになる拡大鍵は、生成しないようにして構わない。この場合には、対応する拡大鍵変換部を備えなくて構わない。

【００９１】このように生成しうる拡大鍵の一部のみをデータ撹拌に使用する構成は、攻撃に対する安全性の面で効果的である。

【００９２】以下、種々のバリエーションについて説明する。

【００９３】まず、データ撹拌部の撹拌処理の段数がｎであり、（拡大鍵変換部を備えたとして）生成可能な拡大鍵の個数がｍ（ｍ＞ｎ）である場合に、基本的には、拡大鍵の重複使用を許さない構成では、ｍ個の拡大鍵のうちからｎ個の拡大鍵を任意に選択したすべての組み合わせが可能である。なお、ここでは、生成される順番に拡大鍵を使用するものとする。

【００９４】なお、拡大鍵の重複使用を許す構成では、基本的には、n^m通りの組み合わせが可能である。

【００９５】いずれの拡大鍵を選択するかについては、ランダムに選択する方法の他に、所定の基準に従って選択する方法がある。

【００９６】ＳＱＵＡＲＥ攻撃と呼ばれる特殊な攻撃では、従来の暗号方式に対して、初段（あるいは最初からの連続する数段）あるいは最終段（あるいは最終段までの連続する数段）の拡大鍵のうち一部のビットに対して、全数探索が行われる。この場合、初段と最終段の拡大鍵が同一である場合には、探索の空間が小さくなってしまい、解読される可能性が高くなる。

【００９７】そこで、初段の拡大鍵変換部により得られる拡大鍵（以下、初段の拡大鍵と呼ぶ）と最終段の拡大鍵変換部により得られる拡大鍵（以下、最終段の拡大鍵と呼ぶ）については、高々一方のみをデータ撹拌に使用するようにしてもよい（いずれか一方のみをデータ撹拌に使用する方法と、いずれもデータ撹拌に使用しない方法とがある）。

【００９８】また、同様に、初段からの連続する数段分の拡大鍵と最終段までの連続する数段分の拡大鍵の範囲において、初段または最終段からの段数を同じくする２つの段の拡大鍵の組について、いずれの組においても、高々一方のみをデータ撹拌に使用するようにしてもよい。この場合に、使用するものあるいは使用しないものについての選択方法には種々のバリエーションが考えられる。例えば、いずれの組においても、いずれか一方を使用するものとする場合に、各組ごとに、使用する方（または使用しない方）を、ランダムに選択してもよいし、例えば前半と後半とから交互になるように選択するなど一定の基準に従って選択するようにしてもよい。また、例えば、各組ごとに、段の順番が前側のものを使用するか、段の順番が後ろ側のものを使用するか、いずれも使用しないかを、ランダムに選択してもよいし、一定の基準に従って選択するようにしてもよい。例えば、図１２に示すように、拡大鍵が１５段分生成可能であり、撹拌処理が９段ある場合に、初段の拡大鍵ｋ１と最終段の拡大鍵ｋ１５のうちではｋ１５を使用し、その１つ内側の段のｋ２とｋ１４のうちからはｋ２を使用し、同様に、ｋ３とｋ１３のうちからはｋ１３を使用し、ｋ４とｋ１２のうちからはｋ４を使用し、ｋ５とｋ１１のうちからはｋ１１を使用し、ｋ６とｋ１０のうちからはｋ６を使用するようにしてもよい。なお、この場合においても、生成される順番に拡大鍵を使用するようにしている。

【００９９】また、初段からの連続する数段分の拡大鍵と最終段までの連続する数段分の拡大鍵の範囲については使用しないようにしてもよい。図１３に、このときの様子を示す。

【０１００】また、初段と最終段、または初段からの連続する数段分の拡大鍵と最終段までの連続する数段分の拡大鍵の範囲については使用せず、その内側の連続する数段分の拡大鍵の範囲については前述と同様に対応する１組のうちの一方のみを使用するようにしてもよい。図１４に、このときの様子を示す。

【０１０１】もちろん、以上の例の他にも、種々のバリエーションがある。

【０１０２】ところで、これまでは、拡大鍵を生成可能な順番で、データ撹拌に使用するものとして説明したが、メモリ等のハードウェアまたは計算時間にある程度の余裕があれば、その余裕に応じて、拡大鍵が生成される順番と、拡大鍵をデータ撹拌に使用する順番とを入れ替えるようにしてもよい。この順番の入れ替えは、図１や図２の構成においても同様である。この順番の入れ替えも、攻撃に対する安全性の面で効果的である。

【０１０３】図１５に、拡大鍵が生成される順番と、拡大鍵をデータ撹拌に使用する順番とを入れ替えた例を示す。

【０１０４】なお、順番を入れ替える場合には、例えば、先に生成された拡大鍵を、後に生成された拡大鍵よりも後で使用するために、一旦、メモリに格納しておくようにすればよい。１つの拡大鍵の順番を入れ替えるだけならば、１つの拡大鍵を一時保存するのに必要なメモリ容量が増加するだけである。

【０１０５】メモリを増加させないためには、中間状態にラウンド関数の逆関数を作用させて必要な中間状態を

復元させればよい。例えば、初段のラウンド関数ｆ１による中間状態kc_1から得られる拡大鍵ｋ１を、２段目のラウンド関数ｆ２による中間状態kc_2から得られる拡大鍵ｋ２を使用した後に使用する場合、一旦、中間状態kc_2を求めた後に、kc_2に２段目のラウンド関数ｆ２の逆関数$f2^{-1}$を作用させて中間状態kc_1を求め（これによって拡大鍵ｋ１が得られる）、さらに中間状態kc_1に２段目のラウンド関数ｆ２を作用させて中間状態kc_2を求め、これに３段目のラウンド関数ｆ３を作用させる…、というようにすることによって、使用順に拡大鍵を生成することができる。なお、ラウンド関数系列がラウンド・トリップ構成を有する場合には、ラウンド関数ｆ２の逆関数$f2^{-1}$も同時に備えているので、これを上記の処理に利用するようにしてもよい。

【０１０６】ところで、以上の拡大鍵の選択や順序の入れ替えは、固定的なものであったが、これを可変とするようにしてもよい。

【０１０７】この場合の図１０／図１１の暗号化装置／復号装置に対応する構成例を、図１６／図１７にそれぞれ示す。図中の７と８はデコーダであり、１５と１６はスイッチング回路である。

【０１０８】この場合、予め各々の攪拌処理Ｒｊへ拡大鍵ｋｉを対応付ける接続パターン（図１２等参照）を複数種類用意しておき、各パターンをコード化し、これを拡張共通鍵ｋｃ’として、本来の共通鍵ｋｃに付加する。

【０１０９】暗号化時には、拡張共通鍵ｋｃ’は、デコーダ７に与えられ、デコーダ７は、拡張共通鍵ｋｃ’を解読し、該拡張共通鍵ｋｃ’が示す接続パターン（例えば、図１２等のパターン）を実現するようにスイッチング回路１５に対するスイッチング制御を行う。これらの動作は、復号時も同様である。

【０１１０】なお、上記では、パターンをコード化するようにしたが、その代わりに、使用しない拡大鍵の段数を示す情報など、他の形態の情報を用いることも可能である。

【０１１１】このような構成も、攻撃に対する安全性の面で効果的である。

【０１１２】なお、以上の各構成例において、図１８に示すように、初段および最終段に擬アダマール変換のような補助関数を挿入するようにしてもよい。この場合に、初段の補助関数と最終段の補助関数に同一の拡大鍵（例えば、初段の拡大鍵）を用いるようにしてもよい。なお、擬アダマール変換は、例えば、ブロック・データの左半分と右半分の算術加算を取ったものを新たな左半分とし、この新たな左半分と右半分の算術加算を取ったものを新たな右半分とするような処理が、これに該当する。

【０１１３】以下では、暗号化装置や復号装置の拡大鍵生成部の複数の拡大鍵変換部のバリエーションについて説明する。

【０１１４】図１９には、１つの拡大鍵生成部の構成例を示す。図１９において、１０１は８ビットのＳ－ｂｏｘであり、１０３はＭＤＳ（Ｍａｘｉｍｕｍ　Ｄｉｓｔａｎｃｅ　Ｓｅｐａｒａｂｌｅ；最大距離分離）行列に基づく３２ビットの攪拌部である。この例は、中間状態の全部または一部として３２×ｋビットのデータを入力し、３２×ｋビットの拡大鍵を出力するもので、４並列のＳ－ｂｏｘ１０１に攪拌部１０３を接続したものを１単位として、これをｋ個分並列に設けたものである。

【０１１５】もちろん、前述したように、拡大鍵生成部は、種々の構成が可能である。

【０１１６】ところで、ある種の暗号解読によりある段（一般には最終段）の拡大鍵（の一部）が知られるおそれは完全には否定できない。かりに、ある段の拡大鍵が知られてしまった場合、拡大鍵変換部の逆変換を行うことで、その段の（ラウンド関数についての）中間状態を知られてしまい、その結果として、他の段の中間状態すべてが知られるところとなり、結局、すべての拡大鍵が知られるところとなるおそれがある。そこで、一部（例えば最終段を含む１段または数段）の拡大鍵変換部においては、逆変換が容易でない関数（例えば、べき乗関数）や、逆が一意に定まらない関数（例えば、多対一関数）を用いるようにしてもよい。これによって、他の段の拡大鍵を容易には知られないようにすることが可能になり、安全性を維持することができる。もちろん、全部の拡大鍵変換部について、逆変換が容易でない関数や、逆が一意に定まらない関数を用いるようにしてもよい。

【０１１７】また、拡大鍵変換部には、対応する段の中間状態の全データを与えてもよいが、その代わりに、対応する段の中間状態の一部のみを渡す構成にして、中間状態の全データを知られないようにし、これによって安全性を維持することもできる。

【０１１８】また、サイドチャネル解析と呼ばれる特殊な攻撃では、ハードウェアで構成された暗号化装置に対して、電力、電磁波等、ＩＣカード等の装置から漏洩する情報をもとに鍵の推測を行う。特に、データ攪拌処理におけるある回路において、同一の構成を有する複数の回路部分があり、それらの回路への入力ビット列とその回路で使用される鍵ビット列（拡大鍵自体または拡大鍵の一部のデータ）が同一であったならば、サイドチャネル情報（例えば、消費電流の変化）の同一性から、それら回路について、入力されたビット列が同一であったことが推測されてしまう。したがって、拡大鍵生成においては、ＩＣカード等で問題となるサイドチャネル解析が容易となるような鍵を生成しないものが望まれる。

【０１１９】そこで、少なくとも処理要素（回路部分）の入出力の一部が直接観測あるいは推定が可能な相異なる処理要素（回路部分）において、同一の拡大鍵が使われないような拡大鍵生成方法が有効である。

【０１２０】なお、全拡大鍵が常には一致しないように、拡大鍵変換部、または拡大鍵変換部およびラウンド処理部、またはラウンド処理部を設計し、偶然一致することは許すようにしてもい。

【０１２１】また、全拡大鍵が常には一致しないように、拡大鍵変換部、または拡大鍵変換部およびラウンド処理部、またはラウンド処理部を設計するとともに、共通鍵の生成時に、全拡大鍵が異なるかどうかをチェックし、全拡大鍵が異なると判定された場合にのみ、その共通鍵を使用するようにしてもよい。

【０１２２】ここで、拡大鍵の一致については、様々なレベルが考えられる。例えば、２つの拡大鍵の全ビットが同一のときに、一致したと判断するようにしてもよい。また、２つの拡大鍵の特定のバイト位置のデータが同一のときには、一致したと判断するようにしてもよい。また、２つの拡大鍵の全ビット、または２つの拡大鍵の特定のバイト位置のデータの間に、一定の関係があるときには、一致したと判断するようにしてもよい。これらの他にも、種々の一致判定方法が可能である。

【０１２３】以下では、暗号化装置や復号装置の拡大鍵生成部の複数のラウンド処理部のバリエーションについて説明する。

【０１２４】図２０には、ラウンド処理部の系列の構成例を示す。図２０の例では、３段分示してあるが、各段の構成が所定段数従属接続されることになる。また、図２０では、共通鍵が１２８ビットであり、各段の拡大鍵は６４ビットである場合を例示している。図中、１０５は非線形写像Ｆであり、１０７の記号は排他的論理和を示している。

【０１２５】非線形写像Ｆは、全段で同一であってもよいし、段ごとに異なっていてもよい。また、後者の場合、基本的には、同じ構成を有するが、段に応じて異なる定数に依存するものであってもよい。

【０１２６】なお、図２０と図２０の逆関数のいずれを暗号化側（または復号側）に用いることも可能である。

【０１２７】一般に、差分解読法や線形解読法といった強力な解読法を用いても、最終段の拡大鍵のうち高々数ビットを特定するのが限界であるから、ラウンド関数の系列は、図２０に示したような単純なＦｅｉｓｔｅｌ構造でも安全性に大きな問題は無いと考えられるが、もし、より強力な解読法の出現に備えてより安全な構造を望むならば、例えば図２１に示すようなラウンド関数の系列を用いてもよい。

【０１２８】図２１の例では、２段分示してあるが、各段の構成が所定段数従属接続されることになる。また、図２１では、共通鍵が１２８ビットであり、各段の拡大鍵は６４ビットである場合を例示している。図中、１０９と１１１と１１３はそれぞれ非線形関数ｆとｇとｈを示しており、１１５の記号は排他的論理和を示している。非線形関数ｆとｇとｈは、すべて同一であっても、すべて異なっても、それらのうち一部のもの同士のみ同一でもよい。

【０１２９】図２２に、非線形関数ｆやｇやｈの構成例を示す。図２２において、１１９は８ビットのＳ－ｂｏｘであり、１２１はＭＤＳ行列に基づく３２ビットの攪拌部である。

【０１３０】図２１では、図２０に比較して、出力される１２８ビットから中間状態を一意に決定することがより困難になっている。

【０１３１】図２３に、図２１の逆関数を示す。なお、図２１と図２３のいずれを暗号化側（または復号側）に用いることも可能である。

【０１３２】以下では、本発明を適用した暗号化装置の一具体例を示す。

【０１３３】図２４に、本暗号化装置の構成例を示す。

【０１３４】この暗号化装置は、１２８ビット（６４ビット）のブロック暗号であり、共通鍵は２５６ビット（１２８ビット）であり、１段の２５６ビット（１２８ビット）である場合を例にとっている。また、ラウンド関数の系列は、ラウンド・トリップ構成を有する場合を例にとっている。また、通常のＳＰＮ構造のＳ－ｂｏｘの部分に小型のＳＰＮ構造を再帰的に埋め込んだ入れ子型ＳＰＮ構造である場合を例にとっている。

【０１３５】図２４において、データ攪拌部では、ラウンド関数（ＤＵ）２０１とラウンド関数（ＤＤ）２０３の繰り返し構造の後、ラウンド関数（ＤＵ）２０１とラウンド関数（ＤＤ（ｗｏＭＤＳＨ））２０５と、ラウンド関数（ＥＸＯＲ）２０７が接続されている。

【０１３６】また、拡大鍵生成部では、ユニット２０９とユニット２１１の対が１段分のラウンド関数に相当する。ただし、図２４の例では、ユニット２０９とユニット２０９の間およびユニット２１１とユニット２１１の間が図１の中間状態ではなく、中間状態はユニット２０９やユニット２１１の内部に現れる構造になっている。

【０１３７】図２５に、１２８ビットのブロック暗号の場合における、図２４のユニット２０１の構成例を示す。図２５において、２１５は鍵加算のための８ビットの排他的論理和であり、２１７は８ビットのＳ－ｂｏｘであり、２１９はＭＤＳ行列に基づく３２ビットの攪拌部である。２１３のユニットが４並列に設けられる。なお、６４ビットのブロック暗号の場合には、２１３のユニットが２並列に設けられる。

【０１３８】図２６に、１２８ビットのブロック暗号の場合における、図２４のユニット２０２の構成例を示す。図２６において、２２１は鍵加算のための８ビットの排他的論理和であり、２２３は１６並列の８ビットのＳ－ｂｏｘであり、２２５はＭＤＳ行列に基づく１２８ビットの攪拌部である。なお、６４ビットのブロック暗号の場合には、２２３のＳ－ｂｏｘが８並列に設けられる。

【０１３９】図２７に、１２８ビットのブロック暗号の場合における、図２４のユニット２０５の構成例を示す。図２７において、２２７は鍵加算のための８ビットの排他的論理和であり、２２９は１６並列の８ビットのＳ－ｂｏｘである。なお、６４ビットのブロック暗号の場合には、２２９のＳ－ｂｏｘが８並列に設けられる。

【０１４０】ユニット２０７は、１２８ビットのブロック暗号の場合、ユニット２０５から出力される１２８ビットのブロック・データに１２８ビットの拡大鍵を加算するための排他的論理和である。また、ユニット２０７は、６４ビットのブロック暗号の場合、ユニット２０５から出力される６４ビットのブロック・データに６４ビットの拡大鍵を加算するための排他的論理和である。

【０１４１】図２８に、共通鍵のビット長が２５６ビットの場合における図２４の拡大鍵生成部の構成例を示す。図２８では、初段部分と折り返し部分のみについて示している。図中、２３１は非線形関数Ｆであり、２３３は排他的論理和であり、２３５は段に依って異なる定数との排他的論理和である。２３７、２３９、２４１、２４３のユニットについては後述する。

【０１４２】図２９に、共通鍵のビット長が１２８ビットの場合における図２４の拡大鍵生成部の構成例を示す。図２９では、初段部分と折り返し部分のみについて示している。図中、２５１は非線形関数Ｆであり、２５３は排他的論理和であり、２５５は段に依って異なる定数との排他的論理和である。２５７、２５９、２６１、２６３のユニットについては後述する。

【０１４３】図３０に、図２８の非線形関数２３１の構成例を示す。図中、２３１１は排他的論理和であり、２３１３はＳ－ｂｏｘであり。２３１５、２３１７については後述する。

【０１４４】次に、図２８のＰ３２と示されたユニット２３７、図２９のＰ１６と示されたユニット２５７、図３０のＰ１６と示されたユニット２３１５、図３０のＰ８と示されたユニット２３１７について説明する。図３１に、これらに共通する一般的な構成例を示す。図中、２６５は排他的論理和であり、あるｉビットと他のｉビットとの排他的論理和を取る操作が４回行われる。この構成をＰｉで表現したものが、各図のＰ８、Ｐ１６、Ｐ３２である。すなわち、図２８のユニット２３７は図３１の構成でｉ＝３２としたものであり、図２９のユニット２５７は図３１の構成でｉ＝１６としたものであり、図３０のユニット２３１５は図３１の構成でｉ＝１６としたものであり、図３０のユニット２３１７は図３１の構成でｉ＝８としたものである。

【０１４５】図３２に、図３１のＰｉの逆変換であるPi^{-1}の構成例を示す。図中、２６７は排他的論理和である。図２８のユニット２４３は図３１の構成でｉ＝３２としたものであり、図２９のユニット２６３は図３１の構成でｉ＝１６としたものである。

【０１４６】なお、図３０は、１２８ビットのブロック暗号の場合であったが、６４ビットのブロック暗号の場合、すなわち図２９の非線形関数２５１の場合には、図３０において、Ｐ８をＰ４にし、Ｐ１６をＰ８にすればよい。

【０１４７】次に、図２８／図２９の５と示されたユニット２３９／２５９、図２９／図３０のＢと示されたユニット２４１／２６１について説明する。図３３に、５と示されたユニットおよびＢと示されたユニットの構成例を示す。両者の違いは、図３３のユニット２６９における関数の内容である。

【０１４８】図３３の構成は、ガロア体ＧＦ（2^4）の元５、またはＢを乗じるものである。

【０１４９】すなわち、入力となる３２ビットを、４組の８ビットに分け、８ビット・データの同じ位置（例えば図３３では最上位ビットの最下位ビットを例にとって示している）の１ビットを集めて、これを１組４ビットのデータとし、８組の４ビット・データの各々を、ＧＦ（2^4）の元とみなす。そして、各々の４ビット・データに対し各々のユニット２６９によって（ガロア体上の乗算に従って）５またはＢを乗じた後に、各々のビットをそれぞれ対応するもとの位置に戻す。

【０１５０】なお、上記では、同じ位置のビットを取り出して処理を行うものとして説明したが、異なる位置のビットを（排他的に）取り出して処理を行うことも可能である。

【０１５１】ガロア体上の乗算は、表引きによっても、演算によっても、実回路によってもよい。

【０１５２】図３４に、図３３のユニット２６９の部分の構成例、すなわちＧＦ（2^4）上の乗算の結線表現（結線パターン）を、元５について（ａ）に、元Ｂについて（ｂ）にそれぞれ示す。なお、前述したように、結合部分２７１では、排他的論理和がなされる。すなわち、この場合、図２８／図２９の５と示されたユニット２３９／２５９については、図３３および図３４（ａ）によって構成可能であり、図２９／図３０のＢと示されたユニット２４１／２６１については、図３３および図３４（ｂ）によって構成可能である。

【０１５３】ところで、データ撹拌部の拡大鍵を作用させる対象が既知あるいは比較的容易に推測可能な部分で使用される拡大鍵、例えばデータ撹拌部の最初の排他的論理和演算や、出力と鍵の推測からデータの推定が可能となる最後の鍵加算の前の排他的論理和演算部への拡大鍵において、異なる位置の演算要素単位（この例の場合、８ビット単位）で拡大鍵が常に一致することや常に一定の関係を持つことを防ぐようにすると好ましい。一構成例としては、上記のデータ撹拌部の最初の排他的論理和演算で使用される拡大鍵と、最後の鍵加算の前の排他的論理和演算部への拡大鍵に要素単位（この例の場合、８ビット単位）で常に成り立つ一致が発生しないよ

うに、拡大鍵を生成する（もしくは共通鍵を選択する）。これにより、サイドチャネル解析を容易ならしめる拡大鍵の一致や一定の関係の成立を妨ぐことができる。

【０１５４】なお、図２４に対応する復号装置の構成は、データ撹拌部については図２４のデータ撹拌部の逆関数になる。また、拡大鍵生成部については、暗号側と復号側ともに最終段のラウンド関数を備えた場合には、図２４の拡大鍵生成部と同様の構成になる。もちろん、暗号側も復号側もそれぞれ最終段のラウンド関数を備えなくてもよい。

【０１５５】なお、図１～図３４を参照しながら説明してきた以上のような実施形態において、１２８ビット等の特定のビット長を例にとったが、もちろん、どのようなビット長のブロックデータでも適用可能である。また、データ撹拌部は、どのような構成であっても適用可能である。

【０１５６】以下では、本実施形態のハードウェア構成、ソフトウェア構成について説明する。

【０１５７】本実施形態の暗号化装置や復号装置は、ハードウェアとしても、ソフトウェアとしても、実現可能である。

【０１５８】本実施形態は、ソフトウェアで実現する場合に、暗号化装置や復号装置を実現するプログラムであって、コンピュータに所定の手段を実行させるための（あるいはコンピュータを所定の手段として機能させるための、あるいはコンピュータに所定の機能を実現させるための）プログラムを記録したコンピュータ読取り可能な記録媒体としても実施することもできる。

【０１５９】また、ハードウェアとして構成する場合、半導体装置として形成することができる。

【０１６０】また、本発明を適用した暗号化装置や復号装置を構成する場合、あるいは暗号化プログラムや復号プログラムを作成する場合に、ブロックもしくはモジュールをすべて個別に作成することも可能であるが、同一構成を有するブロックもしくはモジュールについては１または適当数のみ用意しておいて、それをアルゴリズムの各部分で共有する（使い回す）ことも可能である。

【０１６１】また、ソフトウェアの場合には、マルチプロセッサを利用し、並列処理を行って、処理を高速化することも可能である。

【０１６２】なお、暗号化機能を持ち、復号機能を持たない装置として構成することも、復号機能を持ち、暗号化機能を持たない装置として構成することも、暗号化機能と復号機能の両方を持つ装置として構成することも、可能である。同様に、暗号化機能を持ち、復号機能を持たないプログラムとして構成することも、復号機能を持ち、暗号化機能を持たないプログラムとして構成することも、暗号化機能と復号機能の両方を持つプログラムとして構成することも、可能である。

【０１６３】次に、本実施形態のシステムへの応用について説明する。

【０１６４】本実施形態の暗号方式は、基本的にはどのようなシステムにも適用可能である。

【０１６５】例えば、図３５に示すように、送信側装置３０１と、受信側装置３０３との間で、所定の方法もしくは手続により、鍵を安全に共有しておき、送信側装置３０１は送信データをブロック長ごとに本実施形態の暗号方式で暗号化し、所定のプロトコルに従って、通信ネットワーク３０２を介して、暗号文を受信側装置３０３へ送信し、暗号文を受信した受信側装置３０３では、受信した暗号文をブロック長ごとに本実施形態の暗号方式で復号し、もとの平文を得ることができる。なお、各々の装置が、暗号化機能と復号機能を両方持っていれば、双方向に暗号通信を行うことができる。

【０１６６】また、例えば、図３６に示すように、計算機３１１では、所定の方法で鍵を生成し、保存したいデータをブロック長ごとに本実施形態の暗号方式で暗号化し、所定のネットワーク（例えば、ＬＡＮ、インターネット等）３１４を介して、暗号化データとして、データ・サーバ３１３に保存しておく。計算機３１１では、このデータを読みたいときは、データ・サーバ３１３から所望の暗号化データを読み込み、これをブロック長ごとに本実施形態の暗号方式で復号し、もとの平文を得ることができる。また、他の計算機３１２が、この鍵を知っていれば、同様に復号してもとの平文を得ることができるが、鍵の分からない他の計算機は、該暗号データを復号することはできず、情報のセキュリティ・コントロールが可能になる。

【０１６７】また、例えば、図３７に示すように、コンテンツ提供側では、暗号化装置３２１により、あるコンテンツを、ある鍵で、ブロック長ごとに本実施形態の暗号方式で暗号化し、これを暗号化コンテンツとして、記録媒体３２２に記録し、これを頒布等する。記録媒体３２２を取得したユーザ側では、所定の方法で該ある鍵を入手することにより、復号装置３２３により、該コンテンツを、ブロック長ごとに本実施形態の暗号方式で復号し、コンテンツの閲覧もしくは再生等を行うことができる。

【０１６８】もちろん、上記以外にも種々のシステムに適用可能である。

【０１６９】なお、本実施形態で示した各々の構成は、一例であって、それ以外の構成を排除する趣旨のものではなく、例示した構成の一部を他のもので置き換えたり、例示した構成の一部を省いたり、例示した構成に別の機能を付加したり、それらを組み合わせたりすることなどによって得られる別の構成も可能である。また、例示した構成と論理的に等価な別の構成、例示した構成と論理的に等価な部分を含む別の構成、例示した構成の要部と論理的に等価な別の構成なども可能である。また、

例示した構成と同一もしくは類似の目的を達成する別の構成、例示した構成と同一もしくは類似の効果を奏する別の構成なども可能である。また、各種構成部分についての各種バリエーションは、適宜組み合わせて実施することが可能である。また、本実施形態は、暗号化装置としての発明、復号化装置としての発明、システム全体としての発明、個別装置内部の構成部分についての発明、またはそれらに対応する方法の発明等、種々の観点、段階、概念またはカテゴリに係る発明を包含・内在するものである。従って、この発明の実施の形態に開示した内容からは、例示した構成に限定されることなく発明を抽出することができるものである。

【０１７０】本発明は、上述した実施の形態に限定されるものではなく、その技術的範囲において種々変形して実施することができる。

【０１７１】

【発明の効果】本発明によれば、拡大鍵生成のためのラウンド関数の系列を、共通鍵を入力し、共通鍵と同じ値を出力するように設定することによって、暗号化時と復号時の両方において、従来のような不必要な遅延時間や記憶容量の消費なく、共通鍵からＯｎ－ｔｈｅ－ｆｌｙに拡大鍵を生成することが可能になる。

【図面の簡単な説明】

【図１】本発明の一実施形態に係る暗号化装置の構成例を示す図

【図２】同実施形態に係る復号装置の構成例を示す図

【図３】ラウンド関数の系列の構成について説明するための図

【図４】ラウンド関数の系列の構成について説明するための図

【図５】ラウンド関数の系列の構成について説明するための図

【図６】ラウンド関数の系列の構成について説明するための図

【図７】ラウンド関数の系列の構成について説明するための図

【図８】ラウンド関数の系列の構成について説明するための図

【図９】ラウンド関数の系列の構成について説明するための図

【図１０】同実施形態に係る暗号化装置の他の構成例を示す図

【図１１】同実施形態に係る復号装置の他の構成例を示す図

【図１２】拡大鍵と撹拌処理との接続関係例を示す図

【図１３】拡大鍵と撹拌処理との接続関係例を示す図

【図１４】拡大鍵と撹拌処理との接続関係例を示す図

【図１５】拡大鍵と撹拌処理との接続関係例を示す図

【図１６】同実施形態に係る暗号化装置のさらに他の構成例を示す図

【図１７】同実施形態に係る復号装置のさらに他の構成例を示す図

【図１８】同実施形態に係る暗号化装置のさらに他の構成例を示す図

【図１９】同実施形態に係る拡大鍵生成部の構成例を示す図

【図２０】同実施形態に係るラウンド処理部の構成例を示す図

【図２１】同実施形態に係るラウンド処理部の他の構成例を示す図

【図２２】図２１のラウンド処理部の非線形関数ユニットの構成例を示す図

【図２３】図２１のラウンド処理部の逆関数を持つラウンド処理部の構成例を示す図

【図２４】同実施形態に係る暗号化装置のさらに他の構成例を示す図

【図２５】図２４の第１のユニットＤＵの構成例を示す図

【図２６】図２４の第２のユニットＤＤの構成例を示す図

【図２７】図２４の第３のユニットＤＤ（ｗｏＭＤＳＨ）の構成例を示す図

【図２８】図２４の拡大鍵生成部の構成例を示す図

【図２９】図２４の拡大鍵生成部の他の構成例を示す図

【図３０】図２８と図２９の非線形関数ユニットを説明するための図

【図３１】図２８と図２９の排他的論理和によるユニットを説明するための図

【図３２】図２８と図２９の排他的論理和によるユニットを説明するための図

【図３３】図２８と図２９のガロア体上の乗算によるユニットを説明するための図

【図３４】図２８と図２９のガロア体上の乗算によるユニットを説明するための図

【図３５】同実施形態の暗号方式を利用したシステムの一例を示す図

【図３６】同実施形態の暗号方式を利用したシステムの他の例を示す図

【図３７】同実施形態の暗号方式を利用したシステムのさらに他の例を示す図

【図３８】従来の拡大鍵生成装置について説明するための図

【図３９】従来の拡大鍵生成装置について説明するための図

【符号の説明】

１，２…データ撹拌部

３，４…拡大鍵生成部

１１，１２…撹拌処理部

３１，４２…ラウンド処理部

３３，４４…拡大鍵変換部

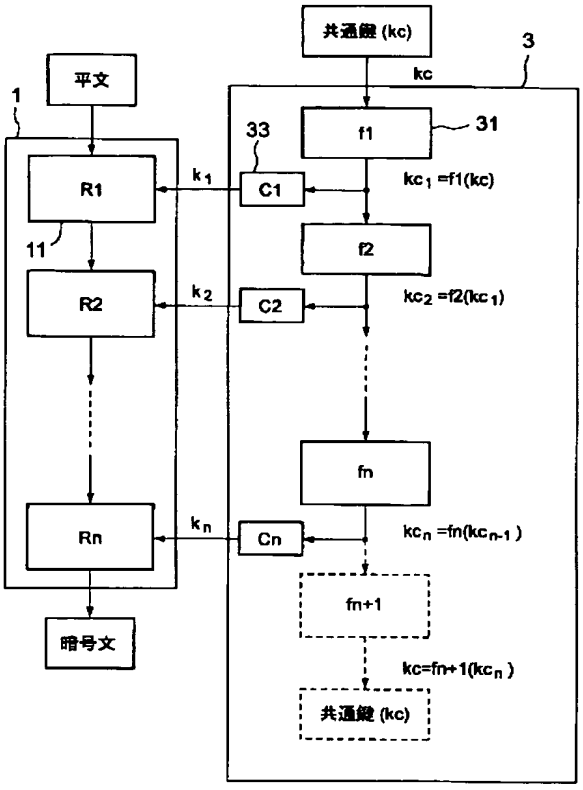
29

7，8…デコーダ

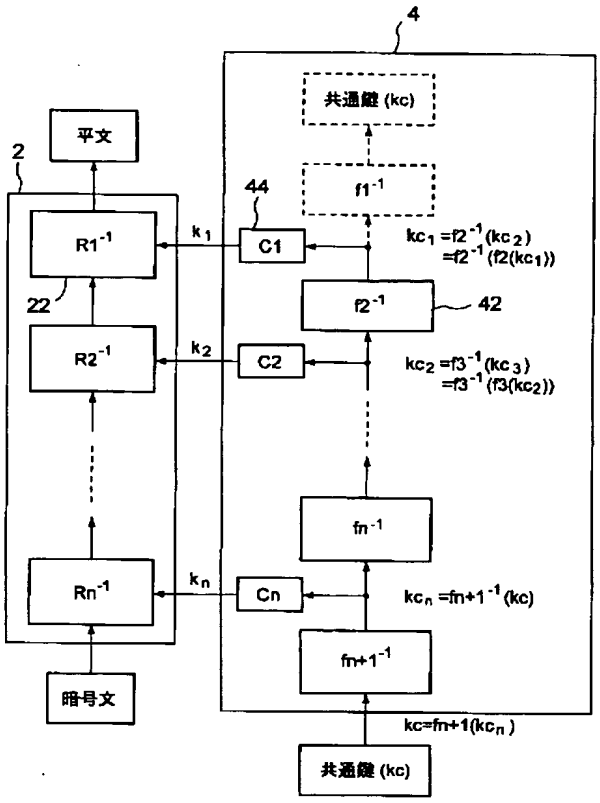
30

１５，１６…スイッチング回路

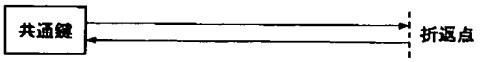
【図１】



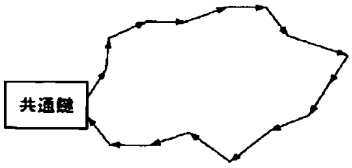
【図２】



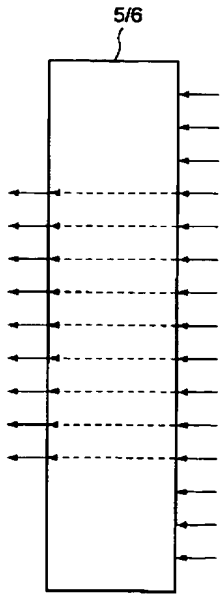
【図３】



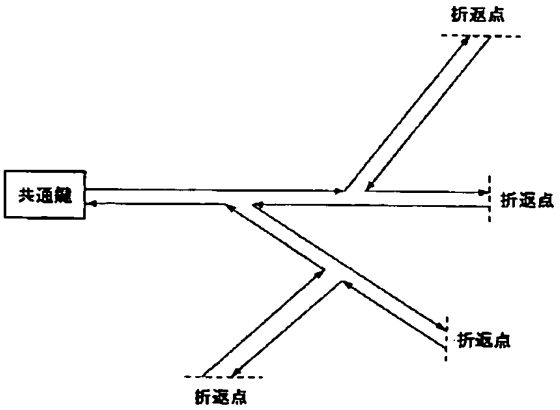
【図４】



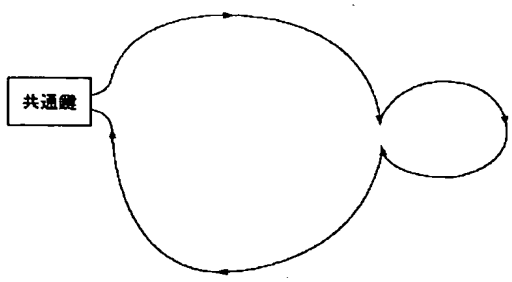
【図１３】



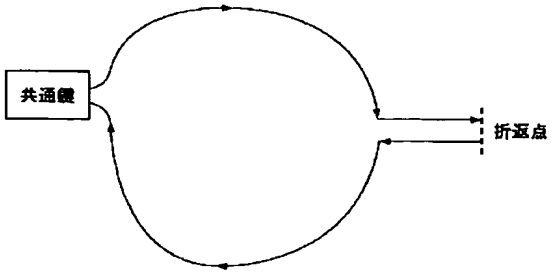
【図５】



【図６】

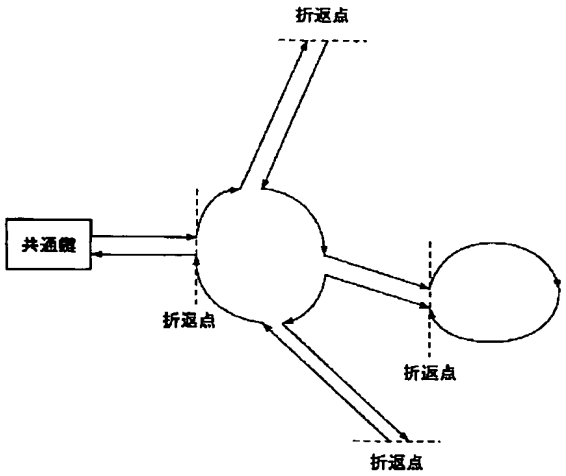


【図７】

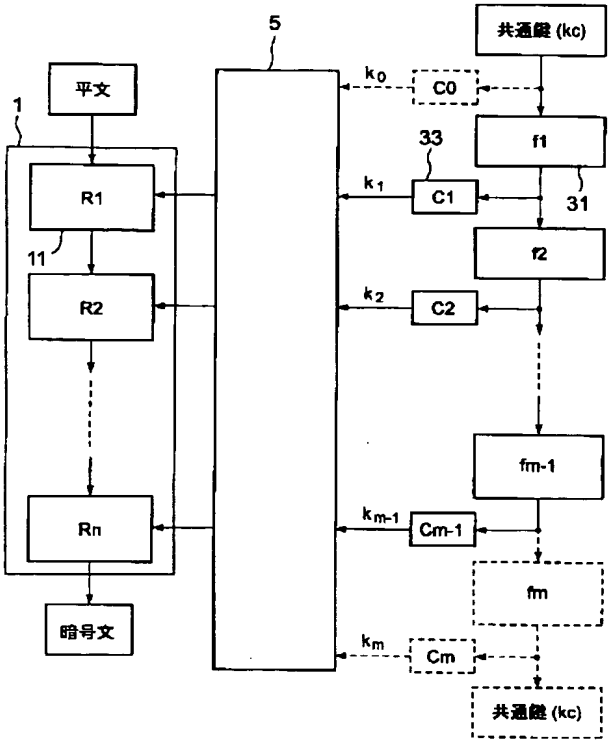


【図８】

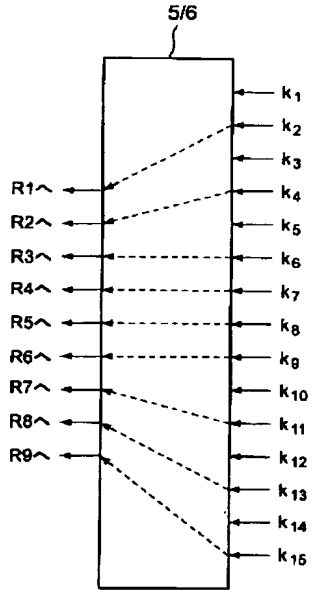
【図９】



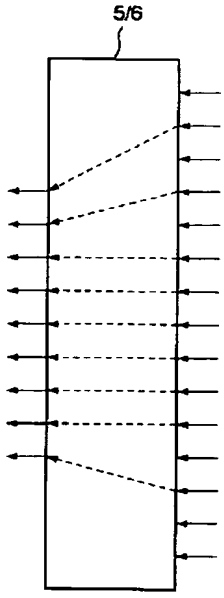
【図１０】



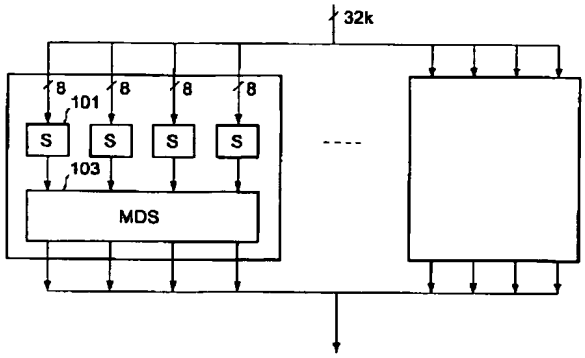
【図１２】



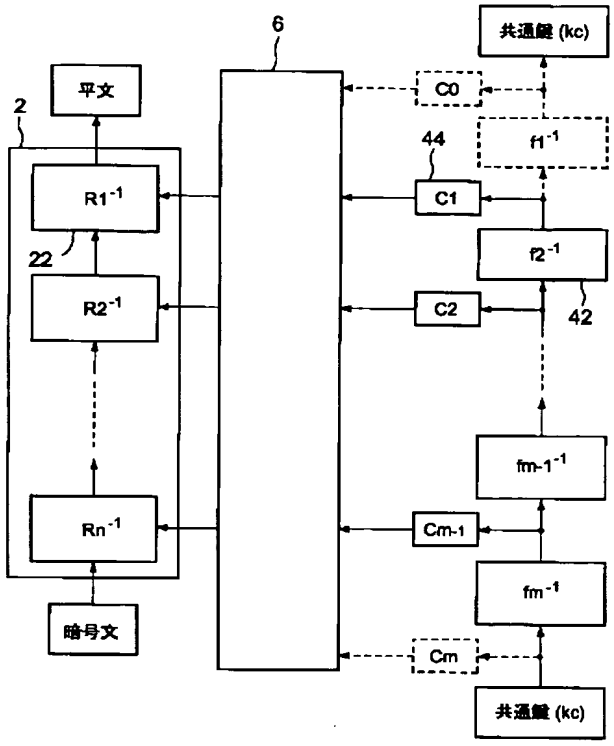
【図１４】



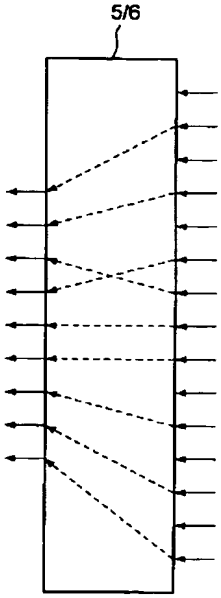
【図１９】



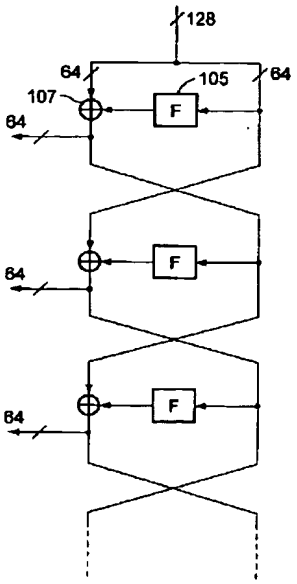
【図１１】



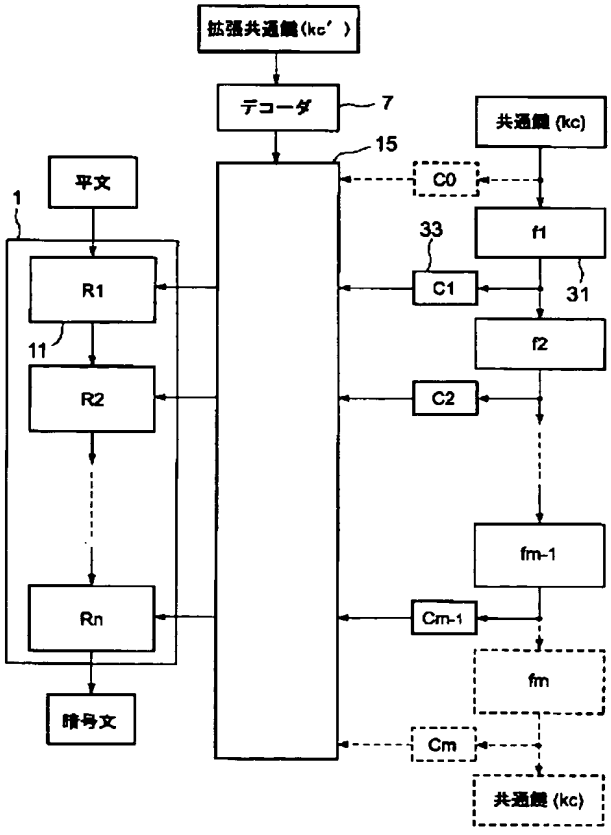
【図１５】



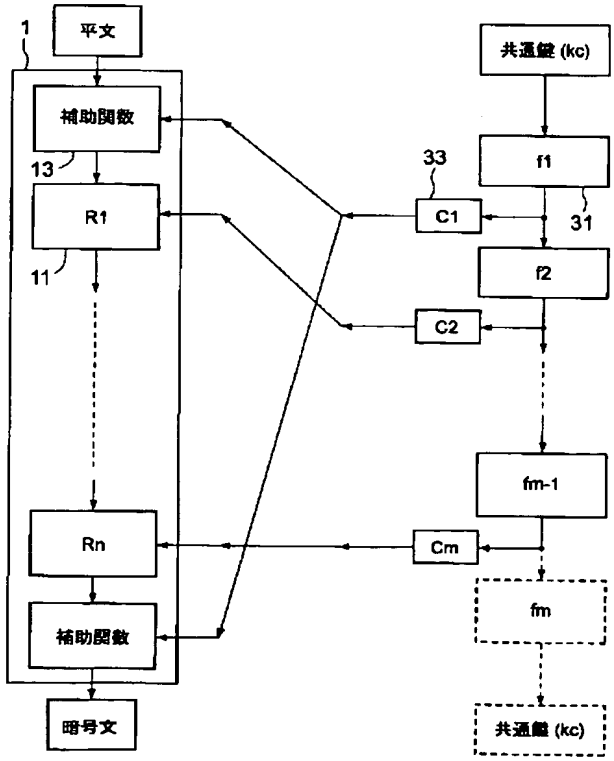
【図２０】



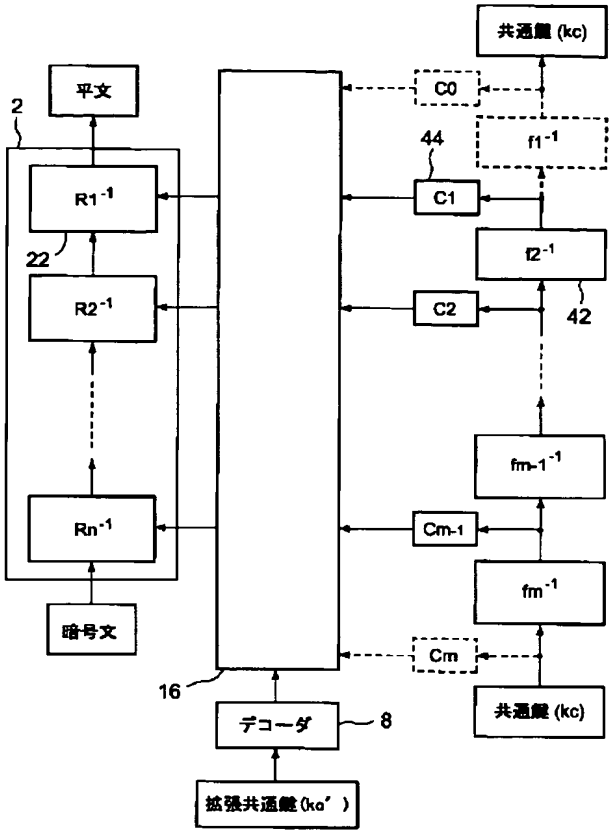
【図１６】



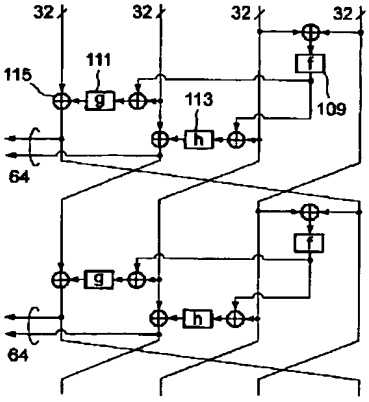
【図１８】



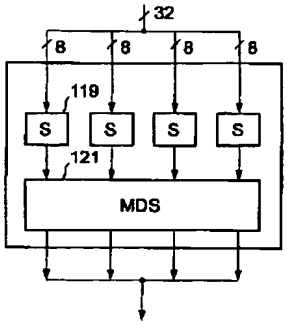
【図１７】



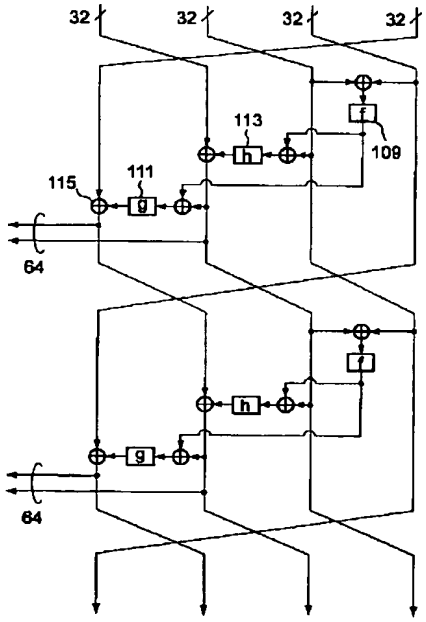
【図２１】



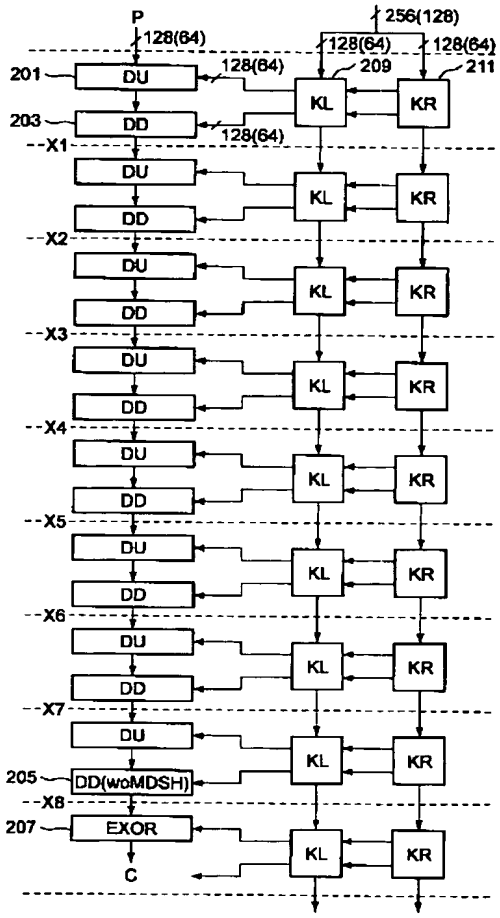
【図２２】



【図２３】

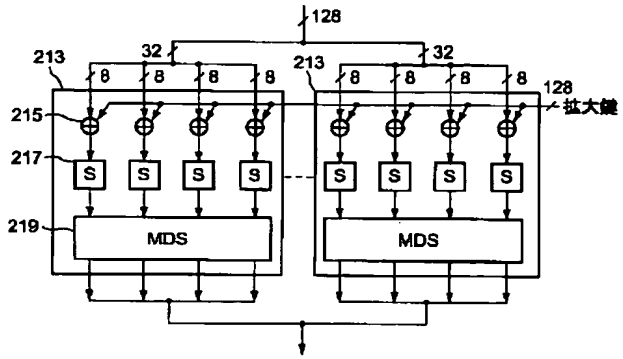


【図２４】

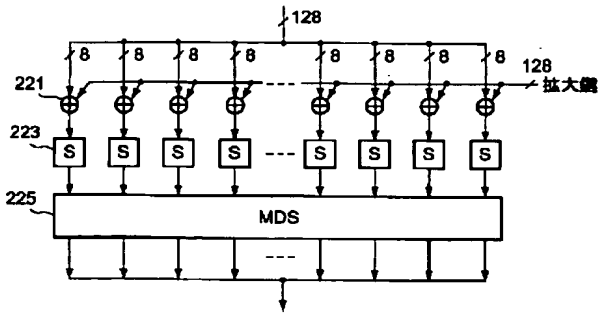


【図３７】

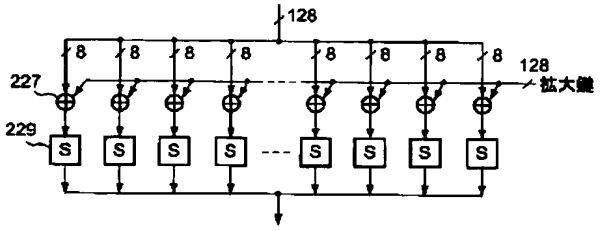
【図２５】



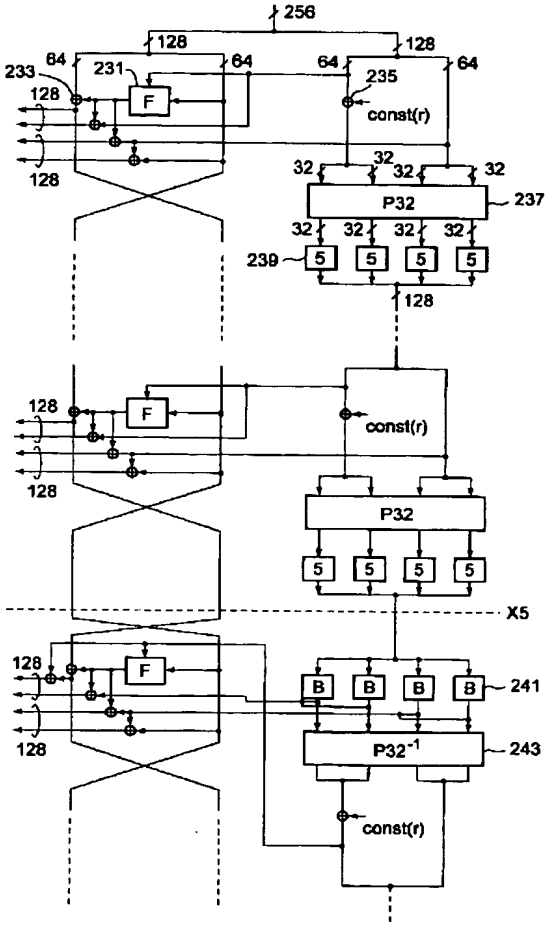
【図２６】



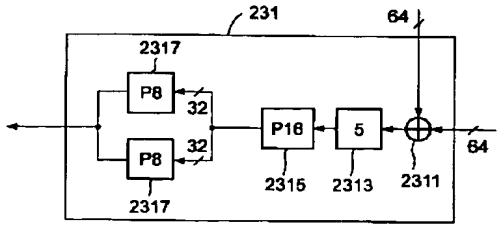
【図２７】



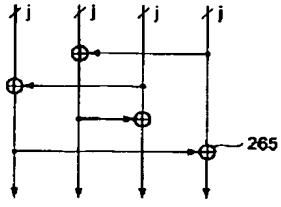
【図２８】



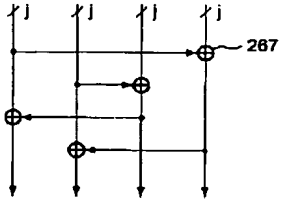
【図３０】



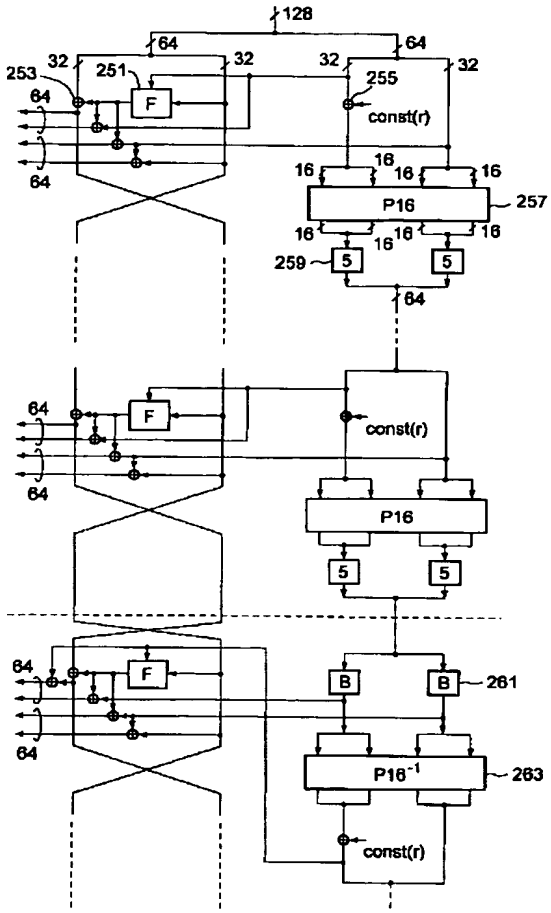
【図３１】



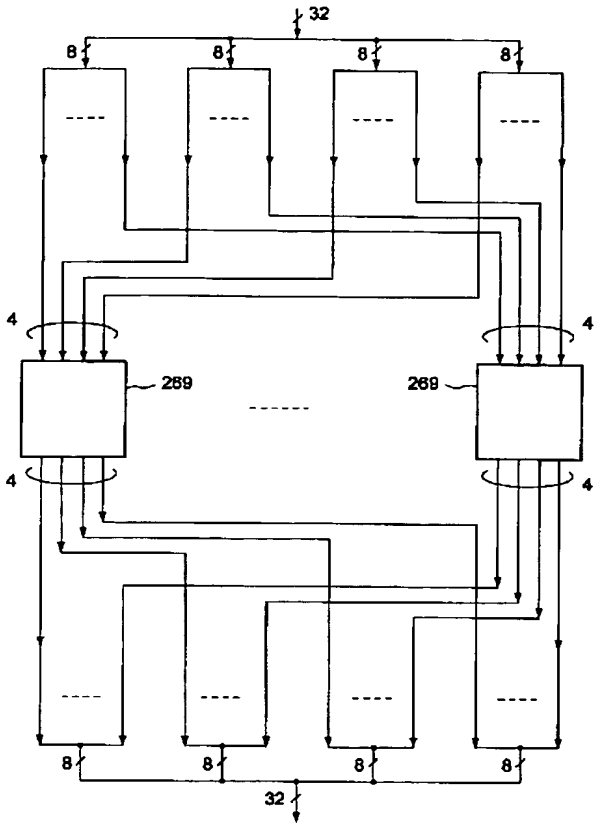
【図３２】



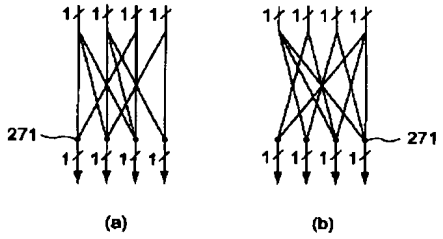
【図２９】



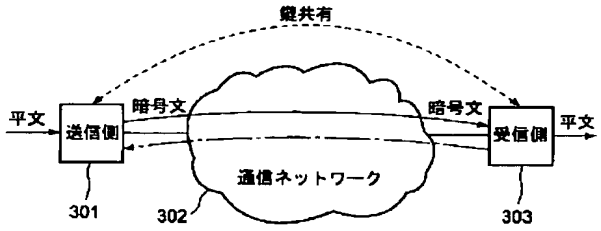
【図３３】



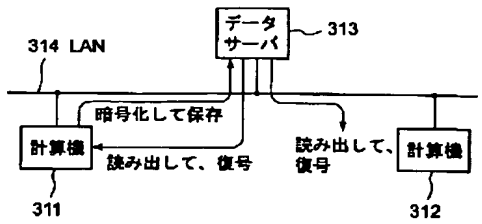
【図３４】



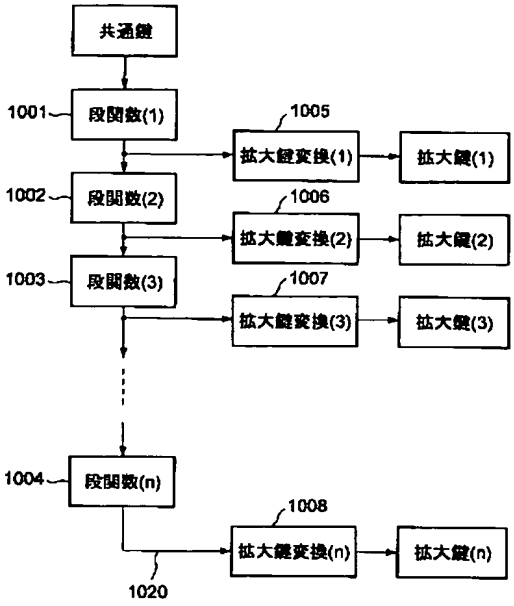
【図３５】



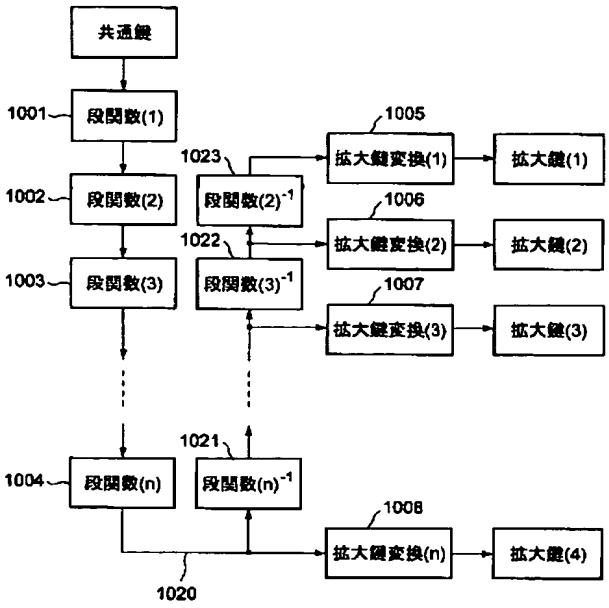
【図３６】



【図３８】



【図３９】



フロントページの続き

(72)発明者　大熊　建司
神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内

(72)発明者　佐野　文彦
東京都府中市東芝町１番地　株式会社東芝府中事業所内

(72)発明者　川村　信一
神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内

Ｆターム(参考)　5J104 JA12 JA15 JA18 NA02 NA08